DEVELOPMENT OF GENETIC ALGORITHM-BASED METHODOLOGY FOR SCHEDULING OF MOBILE ROBOTS

Dang, Vinh Quang

Publication date:
2014

Document Version
Early version, also known as pre-print

Department of Mechanical and Manufacturing Engineering
Aalborg University, Denmark
Special Report No. 98

DEVELOPMENT OF GENETIC ALGORITHM-BASED METHODOLOGY FOR SCHEDULING OF MOBILE ROBOTS

PhD Thesis

by

Quang Vinh Dang

Department of Mechanical and Manufacturing Engineering, Aalborg University
Fibigerstræde 16, DK-9220, Aalborg East, Denmark
e-mail: vinhise@m-tech.aau.dk

Typed in MS Word and printed in Aalborg, January 2014

ISBN: 87-91200-64-4

Mandatory page

1. Thesis title

Development of Genetic Algorithm-Based Methodology for Scheduling of Mobile Robots

2. Name of PhD student

Quang Vinh Dang

3. Name and title of supervisor and any other supervisor

- Supervisor:
  Associate Professor Izabela Ewa Nielsen
  Department of Mechanical and Manufacturing Engineering
  Aalborg University, Denmark
- Co-supervisor:
  Professor Kenn Steger-Jensen
  Department of Mechanical and Manufacturing Engineering
  Aalborg University, Denmark

4. List of published/submitted papers

Papers A and B are reprinted versions of the same paper. Only paper B is included as the printed version.

Papers F and G are reprinted versions of the same paper. Only paper G is included as the printed version.

A. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2011. Scheduling a single mobile robot for feeding tasks in a manufacturing cell. *Proceedings of International Conference Advances in Production Management System*, Stavanger, Norway.

B. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2012. Mathematical formulation for mobile robot scheduling problem in a manufacturing cell. *In*: J. Frick, B. Laugen, eds. *APMS 2011*, *IFIP AICT 384*. Springer-Verlag Berlin Heidelberg, pp. 37–44.

C. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2013. Scheduling a single mobile robot incorporated into production environment. *In*: P. Golinska, eds. *EcoProduction & Logistics*, *EcoProduction.* Springer-Verlag Berlin Heidelberg, pp. 185–201.

D. Dang, Q.V., Nielsen, I., and Bocewicz, G., 2012. A genetic algorithm-based heuristic for part-feeding mobile robot scheduling problem. *In*: J.M.C.

Rodríguez, eds. *Trends in PAAMS, AISC 157*. Springer-Verlag Berlin Heidelberg, pp. 85–92.

E. Dang, Q.V., Nielsen, I., Steger-Jensen, K., and Madsen, O., 2013. Scheduling a single mobile robot for part-feeding tasks of production lines. *Journal of Intelligent Manufacturing*. DOI: 10.1007/s10845-013-0729-y.

F. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2012. Multi-objective mobile robot scheduling problem with dynamic time windows. *Proceedings of International Conference Advances in Production Management Systems*, Rhodes Island, Greece.

G. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2013. Multi-objective genetic algorithm for real-world mobile robot scheduling problem. *In*: C. Emmanouilidis, M. Taisch, D. Kiritsis, eds. *APMS 2012, Part I, IFIP AICT 397*. Springer-Verlag Berlin Heidelberg, pp. 518–525.

H. Dang, Q.V., Nielsen, I., Bøgh, S., and Bocewicz, G., 2013. Modelling and scheduling autonomous mobile robot for a real-world industrial application. *Proceedings of IFAC Conference on Manufacturing Modelling, Management and Control*, Saint Petersburg, Russia, Volume 7, Part 1, pp. 2098-2103.

I. Dang, Q.V., and Nielsen, I., 2013. A methodology for implementation of mobile robot in industrial application. *Robotics & Computer-Integrated Manufacturing* (submitted).

J. Dang, Q.V., and Nielsen, I., 2013. Simultaneous scheduling of machines and mobile robots. *In*: J.M. Corchado et al., eds. *PAAMS 2013 Workshops, CCIS 365*. Springer-Verlag Berlin Heidelberg, pp. 118–128.

K. Dang, Q.V., and Nielsen, I., 2013. Scheduling of machines and mobile robots in FMS. *The International Journal of Advanced Manufacturing Technology* (submitted).

This thesis has been submitted for assessment in partial fulfillment of the PhD degree. The thesis is based on the submitted or published scientific papers which are listed above. Parts of the papers are used directly or indirectly in the extended summary of the thesis. As part of the assessment, co-author statements have been made available to the assessment committee and are also available at the Faculty. The thesis is not in its present form acceptable for open publication but only in limited and closed circulation as copyright may not be ensured.

Abstract

This thesis addresses the issues of scheduling of mobile robot(s) at operational levels of manufacturing systems. More specifically, two problems of scheduling of a single mobile robot with part-feeding tasks and scheduling of multiple mobile robots with preemptive tasks are taken into account. For the first scheduling problem, a single mobile robot is considered to collect and transport container of parts and empty them into machine feeders where needed. A limit on carrying capacity of the single mobile robot and hard time windows of part-feeding tasks are considered. The objective of the first problem is to minimize the total traveling time of the single mobile robot and thereby increase its availability. For the second scheduling problem, a fleet of mobile robots is considered together with a set of machines to carry out different types of tasks, e.g. pre-assembly or quality inspection. Some of the tasks are non-preemptive while the others are preemptive. The considered mobile robots have capabilities to not only transport non-preemptive tasks between some machines but also process preemptive tasks on other machines. These mobile robots are allowed to interrupt their preemptive tasks to carry out transportation of non-preemptive tasks when needed. The objective of the second problem is to minimize the time required to complete all tasks while taking account of precedence constraints.

To deal with each mentioned scheduling problem, each mathematical model is first formulated. This allows describing each problem and finding optimal solutions for each one. However, the formulated mathematical models could only be applicable to small-scale problems in practice due to the significant increase of computation time as the problem size grows. Note that making schedules of mobile robots is part of real-time operations of production managers. Hence to deal with large-scale applications, each heuristic based on genetic algorithms is then developed to find near-optimal solutions within a reasonable computation time for each problem. The quality of these solutions is then compared and evaluated by using the solutions of the mathematical models as reference points. The results from numerical experiments in this thesis show that the proposed heuristics are capable of solving problems of various sizes and more efficient than the mathematical models in terms of the objective values when giving the same limited computation time. The research results are useful for production managers to make decisions at operational levels and the proposed heuristics could be also applied to a variety of tasks of not only mobile robots but also automatic guided vehicles.

Dansk Resumé

Denne ph.d.-afhandling omhandler problemer ved skedulering af mobil(e) robot(ter) på produktionsniveau i produktionssystemer. Mere præcist undersøges to problemer ved skedulering af en enkelt mobil robot med opgaver med fremføring af dele og skedulering af flere mobile robotter med opgaver der kan afbryde en allerede igangsat opgave. Til det første skeduleringsproblem undersøges en enkelt mobil robot. Denne mobile robot skal hente og transportere containere med dele og tømme dem i maskiners fødesystem alt efter hvor der opstår behov for dette. En begrænsning i transportkapaciteten for den enkelte mobile robot og hårde tidsvinduer på processernes eksekvering for fødesystemet for dele undersøges. Målet i første problemstilling er at minimere tidsforbruget for den samlede kørte distance for én mobil robot og derved øge dens kapacitet og oppetid. I det andet skeduleringsproblem undersøges flere mobile robotter sammen med flere maskiner, der udfører forskellige typer af opgaver, fx (for-) montage og kvalitetsinspektion. Nogle opgaver er ikke-afbrydende, mens andre kan afbryde allerede igangsatte aktiviteter. De undersøgte mobile robotter kan ikke kun transportere ikke-afbrydende opgaver mellem maskiner, men kan også udføre proces-afbrydende opgaver på andre maskiner. Disse mobile robotter må afbryde deres opgaver for i stedet at udføre transportopgaver for ikke-afbrydende opgaver, når der opstår behov for dette. Målet med det andet skeduleringsproblem er at minimere tidsforbruget til at færdiggøre alle opgaver, mens der tages højde for forrangs-begrænsninger.

For at undersøge de to nævnte skeduleringsproblemer formuleres først en matematiske model for hvert problem. Dette giver mulighed for at beskrive hvert problem og at finde optimale løsninger for hvert problem. De formulerede matematiske modeller er imidlertid kun anvendelige for virkelige problemer af mindre størrelse, da beregningstiden øges betydeligt i takt med at problemet bliver større. Man skal huske på, at skedulering af mobile robotter foregår som en del af faktisk produktionstid i produktionsplanlægning og -kontrol. Følgelig er det nødvendigt ved anvendelse i forbindelse med større problemer at udvikle hver heuristik baseret på generiske algoritmer for at finde nær-optimale løsninger med fornuftig beregningstid for hvert problem. Kvaliteten af disse løsninger sammenlignes og evalueres så ved at bruge de matematiske modellers løsninger som referencepunkter. Resultaterne fra numeriske eksperimenter i denne ph.d.-afhandling viser, at den foreslåede heuristik kan løse

problemer af forskellig størrelse og gøre det mere effektivt end de matematiske modeller, baseret på de objektive værdier og med den samme begrænsede beregningstid. Forskningsresultaterne er anvendelige for produktionsledere, som kan træffe beslutninger på produktionsniveau; og den foreslåede heuristik kunne også anvendes i forbindelse med andre typer opgaver end mobile robotter, fx for automatisk styrede køretøjer (AGV'er).

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor Izabela Ewa Nielsen for her constant guidance and inspiration during the course of my research and study. Her dedication and enthusiasm have given me great motivation to go through my PhD project period.

Next, I am truly grateful to my co-supervisor Kenn Steger-Jensen for his fruitful advice and encouragement in my work. My sincere thanks are sent to the colleagues and staff at the Department of Mechanical and Manufacturing Engineering, especially Ngoc Anh Dung Do and Peter Nielsen for their valuable discussions and support.

I would also like to thank Associate Professor Shaoping Bai, Associate Professor Jaejin Jang, and Professor Roman Barták for being committee members of my thesis defense.

Furthermore, my special thanks go to the people in the Robotics and Automation Group and people at Grundfos A/S for their collaboration during the EU Tapas project.

Additionally, I am deeply indebted to my parents whose unstoppable love and care have brought me strength and determination to go to the end of the PhD journey where my own belief could not carry me.

My sincere gratitude goes to my girlfriend who shared with me joyfulness and sadness during my past critical years in Aalborg when her sympathy and care have always accompanied me.

Last but not least, I would like to thank all my friends for being with me and supporting me during my difficult times.

Table of Contents

1. Introduction

1.1. Background and motivation

Today's production systems range from fully automated to strictly manual. While the former is very efficient in high volumes but less flexible, the latter has the opposite characteristics. However, manufacturers express a need for transformable production systems that combines the best of both worlds by using new assistive automation and mobile robots (Bischoff, 2010). Mobile robot is a term used to refer to robotic systems consisting of a robot arm mounted on a mobile platform which allow performance of tasks that require both locomotion and manipulation abilities (Hvilshøj et al., 2012a). The concept of mobile robots dates back to 1984 where the MORO was introduced as a robot arm installed on a mobile platform navigating freely on the shop floor, delivering and handling tools and work pieces (Schuler, 1987). Nevertheless, high system costs and lack of processing power prevented its actual use and implementation at that time (Hvilshøj et al., 2012a). Since then a lot of research and development of mobile robots have been carried out, and now the mobile robot technology is on the edge of its breakthrough in the industrial domain. Within this domain, some mobile robots are capable of transporting a variety of part types from one location to another location (Hvilshøj et al., 2012b) similar to material handling devices, e.g. automated guided vehicles (AGVs). In addition, mobile robots have the capabilities to perform more advanced tasks at different machines, workstations, or production lines. These tasks consist of such processes as: machine tending, pre-assembly, and quality inspection (Hvilshøj et al., 2012b). Furthermore, using mobile robots can lead to production efficiency gains, e.g. less energy usage or lower tool-changing costs than conventional industrial robots fixed to one location (Dang et al., 2013a). The superior capabilities of the mobile robots can pave the way for meeting the needs of the transformable production systems.

To utilize mobile robots in an efficient manner requires the ability to properly schedule transporting tasks and more advanced tasks with respect to the needs of manufacturing factories. Therefore, it is important for scheduling to determine in which sequences the mobile robots should process those tasks so that they could effectively work while satisfying a number of technological constraints.

Scheduling is a decision-making process that is used on a regular basis in many manufacturing and services industries. It deals with the allocation of resources to tasks over given time periods and its goal is to optimize one or more objectives (Pinedo, 2008). The resources and tasks in a manufacturing factory can take many different forms. The resources may be machines, tools, material handling devices, and so on in a workshop. The tasks may be operations in a production process. Each task may have a certain priority level, release date, and due date. The objectives can also take many different forms, for instances, the minimization of the completion time of the last task, or the minimization of traveling time of material handling devices. In a manufacturing environment, the scheduling function often interacts with other systems, e.g. Enterprise Resource Planning (ERP) systems, or other functions, e.g. shop floor management functions (Pinedo, 2008) as depicted in Figure 1.1. The scheduling function receives daily plans from the ERP systems, makes detailed schedules under consideration of information given in the daily plans, and puts the detailed schedules into action through the shop floor management functions. In other words, the scheduling function has a major impact on these systems or functions and vice versa. Therefore, scheduling as a decision-making process plays an important role in most of manufacturing and production systems and the development of a detailed task schedule helps maintain efficiency and control of operations.

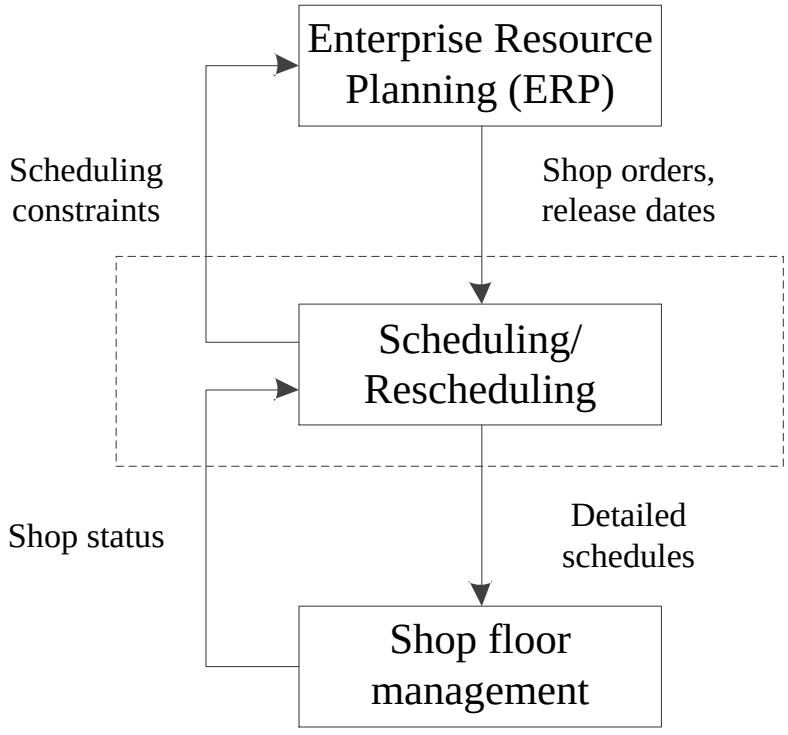

Figure 1.1: Interaction of scheduling function with other systems (Pinedo, 2008)

Within the scope of the thesis, the problems of scheduling of a single mobile robot and scheduling of multiple mobile robots at operational levels of manufacturing systems are considered. For the first scheduling problem, a single mobile robot with a manipulation arm is considered to perform part-feeding tasks which are the process of loading several components at a time into feeders. The single mobile robot during operation has the capability to collect and transport containers of parts and empty them into feeders where needed (Hvilshøj et al., 2012a). The limit on carrying capacity of the single mobile robot and the hard time windows (Toth and Vigo, 2002) of part-feeding tasks are simultaneously considered in the first problem. Because of these constraints, the single mobile robot has to serve a set of feeders in more than one route during a given planning horizon while still meeting the time windows constraints. This means that the number of routes and the sequence which the single mobile robot travels these have to be determined in order to minimize its total traveling time and thereby increase its availability. Following the single mobile robot case, it is possible to extend the research by taking into account multiple mobile robots which help to serve and satisfy more production needs. Hence, for the second scheduling problem, a set of mobile robots are considered together with a set of machines to execute a number of operations of different tasks, e.g. pre-assembly or quality inspection. Some of the tasks are non-preemptive while the others are preemptive. It means that operations of non-preemptive tasks must execute without interruption from its starting time to its ending time while operations of preemptive tasks can be interrupted at any time (Yun, 2002; Duša and Barták, 2009). During operation, the considered mobile robots have the capabilities to not only transport non-preemptive tasks between some machines but also process preemptive tasks on other machines by using their manipulation arms. These mobile robots are allowed to interrupt their preemptive tasks to perform transportation of non-preemptive tasks when needed. The objective of the second is to minimize the time required to complete all tasks, i.e. makespan while considering precedence constraints.

So far there have been done a number of researches related to the class of mobile robot scheduling problems. However, these related researches have not focused on the area of the two described problems. The main novelties of the thesis lie on the fact that: (i) the first problem simultaneously considers hard time windows of part-feeding tasks and multiple delivery routes in case of the single mobile robot; (ii) the second problem takes account of simultaneous scheduling of machines and multiple mobile robots with preemptive tasks. The surveyed approaches are not well suited and cannot be directly

used to solve the two described problems due to the lack of efficient mechanisms to schedule mobile robot(s) with the aforementioned considerations. Hence in the thesis, each mathematical model is first formulated which allows describing each problem as well as finding optimal solutions for each one. Note that the first problem can be considered as a variant of the Asymmetric Traveling Salesman Problem (Reinelt, 1991) which belongs to the NP-hard class (Germs et al., 2012). Moreover, the second problem can be considered as a variant of the problem of simultaneous scheduling of machines and AGVs in which its sub-problems, machine scheduling and AGV scheduling, are both known to be NP-hard (Deroussi et al., 2008). Due to the intractability of NP-hard nature (Ganesharajah et al., 1998), mathematical approaches could only be applicable to small-scale problems in practice because their computation time significantly increases when the problem size grows (Gen and Lin, 2008). Note that making schedules of mobile robots is part of real-time operations of production managers. Therefore, in order to deal with large-scale applications, each heuristic based on genetic algorithms (GA) (Goldberg, 1989) is then developed to find near-optimal solutions for each described problem. GA is a promising algorithm for the class of the described problems. In GA, each individual solution is represented in the form of a finite length string called a chromosome. A chromosome is composed of a set of locations known as genes that contain discrete values pertaining to a problem solution. Through the use of genetic operators such as crossover, mutation, and selection to the chromosomes of selected solutions are in a systematic fashion to generate a new generation of solutions moving towards the optimization of certain criteria (Gen and Lin, 2008). Compared to other optimization methods, the major benefit of GA regards multiple directional searches using a set or population of candidate solutions which enables GA to search in several directions concurrently. In this way, many paths to the optimum are processed in parallel leading to a clear improvement in performance. Since information from many different regions is used, GA is also resistant to remain trapped in a suboptimal solution and able to move away from it if the population finds better solutions in other search areas (Dang and Nielsen, 2013). With these advantages, GA seems to be a proper method to find efficient solutions for the described problems. Finally, for each described problem, the comparisons between the proposed heuristic and mathematical model in terms of objective value and computational time are conducted. These comparisons help to evaluate the quality of the near-optimal solutions achieved by the proposed heuristic by using solutions of the mathematical model as reference points.

1.2. Structure of the thesis

The thesis consists of two distinct parts. The first part of the thesis presents the research results achieved through the PhD project. This part is structured as follows. First, a problem of scheduling a single mobile robot for part-feeding tasks of production lines is considered (Chapter 2). Second, a problem of simultaneously scheduling of machines and multiple mobile robots in a manufacturing system under consideration of some preemptive tasks is taken into account (Chapter 3). The common theme of these chapters is to describe problems, formulate a mathematical model and develop a heuristic based on genetic algorithms to solve problems, and conduct computational experiments to demonstrate and compare the performance of the proposed approaches. Finally, concluding remarks and a discussion of future works into the research area are presented (Chapter 4). The second part of the thesis is a collection of papers submitted and published as a part of the research. This part contains nine selected papers.

1.3. Publications and submission during PhD study

The papers published and submitted during PhD period including the main contributions

Papers A and B are reprinted versions of the same paper. Only paper B is included as the printed version.

Papers F and G are reprinted versions of the same paper. Only paper G is included as the printed version.

A. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2011. Scheduling a single mobile robot for feeding tasks in a manufacturing cell. *Proceedings of International Conference Advances in Production Management System*, Stavanger, Norway.

B. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2012. Mathematical formulation for mobile robot scheduling problem in a manufacturing cell. *In*: J. Frick, B. Laugen, eds. *APMS 2011*, *IFIP AICT 384*. Springer-Verlag Berlin Heidelberg, pp. 37–44.

C. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2013. Scheduling a single mobile robot incorporated into production environment. *In*: P. Golinska, eds. *EcoProduction & Logistics, EcoProduction.* Springer-Verlag Berlin Heidelberg, pp. 185–201.

D. Dang, Q.V., Nielsen, I., and Bocewicz, G., 2012. A genetic algorithm-based heuristic for part-feeding mobile robot scheduling problem. *In*: J.M.C.

Rodríguez, eds. *Trends in PAAMS, AISC 157*. Springer-Verlag Berlin Heidelberg, pp. 85–92.

E. Dang, Q.V., Nielsen, I., Steger-Jensen, K., and Madsen, O., 2013. Scheduling a single mobile robot for part-feeding tasks of production lines. *Journal of Intelligent Manufacturing*. DOI: 10.1007/s10845-013-0729-y.

F. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2012. Multi-objective mobile robot scheduling problem with dynamic time windows. *Proceedings of International Conference Advances in Production Management Systems*, Rhodes Island, Greece.

G. Dang, Q.V., Nielsen, I., and Steger-Jensen, K., 2013. Multi-objective genetic algorithm for real-world mobile robot scheduling problem. *In*: C. Emmanouilidis, M. Taisch, D. Kiritsis, eds. *APMS 2012, Part I, IFIP AICT 397*. Springer-Verlag Berlin Heidelberg, pp. 518–525.

H. Dang, Q.V., Nielsen, I., Bøgh, S., and Bocewicz, G., 2013. Modelling and scheduling autonomous mobile robot for a real-world industrial application. *Proceedings of IFAC Conference on Manufacturing Modelling, Management and Control*, Saint Petersburg, Russia, Volume 7, Part 1, pp. 2098-2103.

I. Dang, Q.V., and Nielsen, I., 2013. A methodology for implementation of mobile robot in industrial application. *Robotics & Computer-Integrated Manufacturing* (submitted).

J. Dang, Q.V., and Nielsen, I., 2013. Simultaneous scheduling of machines and mobile robots. *In*: J.M. Corchado et al., eds. *PAAMS 2013 Workshops, CCIS 365*. Springer-Verlag Berlin Heidelberg, pp. 118–128.

K. Dang, Q.V., and Nielsen, I., 2013. Scheduling of machines and mobile robots in FMS. *The International Journal of Advanced Manufacturing Technology* (submitted).

Additional research and contributions completed during PhD period

- Dang, Q.V., Nielsen, I., and Yun, W.Y., 2013. Replenishment policies for empty containers in an inland multi-depot system. *Maritime Economics & Logistics*, Volume 15, Issue 1, pp. 120–149.
- Bocewicz, G., Nielsen, P., Banaszak, Z., and Dang, Q.V., 2013. Multimodal Processes Cyclic Steady States Scheduling. *In*: J.M. Corchado et al., eds. *PAAMS 2013 Workshops, CCIS 365*. Springer-Verlag Berlin Heidelberg, pp. 73–85.
- Bocewicz, G., Banaszak, Z.A., Nielsen P., and Dang, Q.V., 2013. Multimodal processes rescheduling. *In*: C. Emmanouilidis, M. Taisch, D. Kiritsis, eds. *APMS 2012, Part I, IFIP AICT 397*. Springer-Verlag Berlin Heidelberg, pp. 534–541.

- Nielsen, I., Nielsen, P., and Dang, Q.V., 2012. Decision support for multi-criteria project portfolio evaluation. *Journal of Operations and Logistics* (under review).
- Bocewicz, G., Nielsen P., Banaszak, Z.A., and Dang, Q.V., 2012. Cyclic steady state refinement: multimodal processes perspective. *In*: J. Frick, B.T. Laugen, eds. *APMS 2011, IFIP AICT 384*. Springer-Verlag Berlin Heidelberg, pp. 18–26.
- Bocewicz, G., Banaszak, Z.A., Nielsen P., and Dang, Q.V., 2012. Multimodal processes rescheduling. *Proceedings of International Conference Advances in Production Management Systems*, Rhodes Island, Greece.
- Bocewicz, G., Nielsen P., Banaszak, Z.A., and Dang, Q.V., 2011. Cyclic steady state refinement: multimodal processes perspective. *Proceedings of International Conference Advances in Production Management Systems*, Stavanger, Norway.
- Dang, Q.V., Nielsen, I., and Yun, W.Y., 2011. Policies for positioning empty containers in an inland multi-depot system. *Proceedings of International Conference on Industrial Engineering and System Management*, Metz, France, pp. 66–75.

2. Single mobile robot with part-feeding tasks

2.1. Introduction

In this chapter, a problem of scheduling of a single mobile robot which performs part-feeding tasks of production lines is addressed. Part-feeding is one of the most suitable industrial applications in which the mobile robot collects and transports containers of parts and empties them into the feeders. The mobile robot has to be scheduled in order to prevent stoppage due to lack of parts in the production lines. A method based on the characteristics of feeders and inspired by the (s, Q) inventory system (Silver et al., 1998) is thus applied to determine time windows for the part-feeding tasks of the mobile robot. The capacity of the mobile robot, which is a limited number of containers of parts the mobile robot can carry at a time, is also considered. The performance criterion is to minimize the total traveling time of the mobile robot for a given planning horizon and thereby increase its availability. Note that making decision on which sequence the mobile robot should perform tasks is part of real-time operations. This gives the added requirement that the best sequences of tasks must be obtained quickly. Moreover, the complexity of the problem rapidly rises as the mobile robot has to serve more feeders and/or work in a longer planning horizon. Therefore, in this chapter the focus is on developing a computationally efficient approach, namely a GA-based heuristic for scheduling part-feeding tasks of the mobile robot. A mixed-integer programming (MIP) model is also developed to assess the performance of the proposed heuristic. In the next section, the literature survey will be carried out as part of theoretical foundations relating to this research.

2.2. Survey of literature

The problem of scheduling part-feeding tasks of the mobile robot has been modeled in several respects comparable to the Traveling Salesman Problem (TSP) or Asymmetric Traveling Salesman Problem (ATSP) (Reinelt, 1991). However, the problem is similar to but not identical to the TSP or ATSP due to some additional constraints which the problem possesses. Several approaches and models for exact or heuristic algorithms have been proposed to address problems of this type. Carpaneto and Toth (1980) present a branch-and-bound algorithm for the ATSP based on the sub-tour elimination approach

or the Hungarian algorithm. They discuss a new selection procedure for the sub-tour to be split and the ordering of the arcs in the selected sub-tour. A similar approach is presented by Syslo et al. (1983). This approach is based on the Hungarian algorithm and reduces the original distance matrix till an optimal solution is obtained. It is shown that the execution time is strongly dependent on the problem instance and increases with the size of the network. Miller and Pekny (1991) survey methods such as branch-and-bound and several heuristics for solving large TSP problems. A branch-and-bound algorithm is presented with computational results and is found to perform well for some classes of problems. The branch-and-bound methods (Carpaneto et al., 1995) for the ATSP use the assignment problem as a relaxation. The effectiveness of the methods derives from reduction procedures and parametric solution of the relaxed problems associated with the nodes of the branch decision tree. Turkensteen et al. (2008) introduce a branch-and-bound algorithm for the ATSP using the upper tolerances values of arcs in the corresponding assignment problem instance to determine which arcs should be excluded. The class of tolerance-based algorithms is better in solving difficult instances than the algorithm presented in e.g. Carpaneto et al. (1995). Germs et al. (2012) then enhance this approach by incorporating lower tolerances, corresponding to additional costs of a solution with a connecting arc, into the branch-and-bound search process. Ascheuer et al. (1993) present a cutting plane approach to the sequential ordering problem which is similar to the robot task-sequencing problem and find minimum cost paths subject to precedence constraints. They outline a Linear Programming framework and discuss polynomial time separation algorithms for obtaining the solutions. The problem of order-picking in a rectangular warehouse of Automated Storage and Retrieval System is addressed by Ratliff and Rosenthal (1983). It is shown to be a solvable case of the TSP and they present an algorithm for picking an order in minimum time. Edan et al. (1991) present a near-minimum task-planning algorithm for a fruit harvesting robot to find near-optimal-time path between the N given fruit locations. The sequence of motions for the harvesting robot is obtained by solving the TSP using the geodesic distance. Dang et al. (2012) propose an MIP model to obtain the optimal feeding sequence of a mobile robot in a manufacturing cell. However, the performance of the MIP model is not evaluated and compared with other methods. For small task-scheduling problems, the aforementioned techniques can be used to find the optimal solutions of the problems. Nevertheless, they tend to get computationally intractable for large and complex problems (Maimon et al., 2000).

Larger problems call for heuristic solutions (Maimon et al., 2000). The heuristic approaches that are frequently applied to robot task-scheduling problems include: the nearest-neighbor rule in which the robot travels to the nearest pickup point from its current position (Han et al., 1987), the closest insertion algorithm which causes the smallest increase in the length of the sequence (Askin and Stanridge, 1993), and dispatching rules (e.g. First-Come-First-Served) which serve the tasks in chronological order (Suárez and Rosell, 2005). The above listed heuristics have shown good results and are computationally fast. Another set of heuristic solutions including ant colony optimization, simulated annealing, tabu search, neural network and genetic algorithm has been used to solve combinatorial optimization problem such as TSP, ATSP or robot task-scheduling problems. Tsai et al. (2004) discuss using ant colony system to solve the TSP, while López-Ibáñez and Blum (2010) apply the ant colony optimization technique to deal with the TSP with time windows (TSPTW). Ohlmann and Thomas (2007) describe a variant of simulated annealing incorporating a variable penalty method to solve the TSPTW, while Geng et al. (2011) deal with the TSP based on an adaptive simulated annealing with greedy search. Carlton and Barnes (1996) present a robust tabu search approach to the TSPTW. Landrieu et al. (2001) present tabu search and probabilistic tabu search for single vehicle pickup and delivery problem. Hurink and Knust (2002) propose a tabu search algorithm for scheduling a single robot in a job-shop environment. Hasegawa et al. (2002) develop two types of searching methods based on tabu search and neural networks for solving large scale TSPs. Maimon et al. (2000) present a neural network approach successfully implementing the robot task-sequencing problem. Among the heuristics proposed in previous work, genetic algorithms (GA) have been widely used in solving TSP or ATSP because GAs have a global search ability and can easily be implemented (Chen and Chien 2011). Chatterjee et al. (1996) propose a GA with an asexual reproduction plan through a generalized mutation operator that can be applied to TSP. Moon et al. (2002) present an efficient GA with a topological sort and a new crossover operation to solve the TSP with precedence constraints. Snyder and Daskin (2006) combine a GA with a local tour improvement heuristic and encoded solution using random keys for solving the generalized TSP. Liu and Zheng (2009) present an improved GA with reinforcement mutation to solve the TSP. Choi et al. (2003) propose a GA that extends the search space by purposefully generating and including infeasible solutions to solve ATSP. Xing et al. (2008) present a novel hybrid approach incorporating a GA improved on

both crossover and mutation operators and some optimization strategies such as immigration, local and global optimization for the ATSP. Chen and Tseng (1996) and Zacharia and Aspragathos (2005) introduce methods based on GAs and innovative encoding to determine the optimal sequence of robot's task points that is considered an extension to the TSP. Dang et al. (2013a) present the bartender concept to incorporate a mobile robot into production and discuss the problem of scheduling the mobile robot under consideration of environmental consciousness concerning the usage of battery energy on transportation on the shop floor.

There have been carried out a number of researches related to the class of mobile robot scheduling problem. However, little attention is paid to the problem of scheduling a single mobile robot with time windows, restricted capacity and multiple tasks having to be carried out during a planning horizon despite its important application in practice, e.g. part-feeding tasks. In this problem, a number of tasks with time windows have to be satisfied by the mobile robot. Nevertheless, due to the limit on carrying capacity, after satisfying some tasks, the robot has to return to a warehouse (base) to load parts so that it can serve other tasks in the next route and so on (a route is from the warehouse, to locations of tasks, and back to the warehouse). In other words, the robot has to travel on a number of routes to perform tasks. The surveyed approaches are not well suited and cannot be directly used to solve this problem due to the lack of a mechanism handling both time windows and multiple delivery routes in case of the single mobile robot. It is thus necessary to develop a proper and efficient method to deal with this problem.

2.3. Problem Description

The work is developed for a real cell that produces parts for the pump manufacturing industry at a factory in Denmark. Figure 2.1 shows a typical layout of a manufacturing cell in which the interaction between a mobile robot and machines on production lines is taken into account.

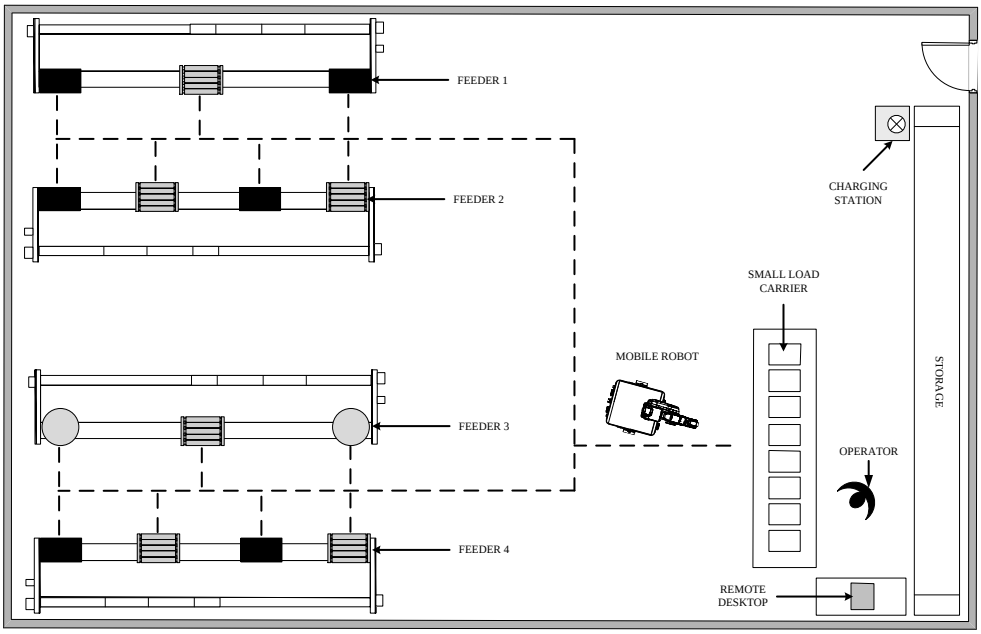

Figure 2.1: Layout of a manufacturing cell

Before assigning the mobile robot to the production environment, the manufacturing cell has one or several production lines which consist(s) of multiple machines. Feeders are designed to automatically supply parts to these machines. Pallets or boxes, which contain the parts, are placed next to these feeders. Part-feeding, the process of loading many parts at a time into feeders from the pallets or boxes, is a manually performed, non-value adding manufacturing task, and quite often disruptive (in-between and periodic) for production workers. Furthermore, when workers forget to fill the feeders, this may lead to stopping the production lines. A strategy that can reduce the dependence on human intervention for the part-feeding tasks is using a mobile robot instead of humans. However, to utilize the robot in this scenario requires changing the work environment and carefully scheduling part-feeding tasks of the mobile robot (Dang et al., 2013a).

To meet the stated requirement, the bartender concept (Hvilshøj et al., 2012b) is implemented. This concept helps to structure the way how the mobile robot carries out the part-feeding tasks in the production environment. This concept also serves as a basis to model the problem of scheduling part-feeding tasks of the mobile robot and can be applied to different sizes of the problem. In this concept, every feeder is assigned the following characteristics: maximum level, minimum level, and feeding time per part to machine. Furthermore, instead of scattered pallets or boxes containing parts next to the feeders, a central warehouse (a bar) is created to gather different parts into one area. An

operator (the bartender) put parts into small load carriers (SLCs) which are placed in the warehouse. The number of parts inside each SLC is equal to the difference between maximum level and minimum level of parts of the feeder in which that SLC is emptied. During operation, the mobile robot will retrieve and carry one or several SLCs containing parts from the warehouse, move to feeder locations, empty all parts inside SLCs, return to the warehouse to unload all empty SLCs and take new filled SLCs. Each feeder has to be served a number of times in order to keep the production line in operation. The mobile robot thus has a set of subtasks possessing time windows to carry out for each feeder during a planning horizon. In order to accomplish all the movements with the smallest consumed amount of the battery energy and thereby increase the availability of the mobile robot, the total traveling time of the mobile robot must be considered. Note that making decisions on which sequences the mobile robot should supply parts to the feeders is part of the real-time operations of production planners. It means that the best solution must be quickly obtained at the beginning of production shifts or during the shifts due to some errors in a manufacturing cell (e.g. machine breakdown) or changes in a manufacturing cell's conditions (e.g. cycle time of production lines). Moreover, as the problem is NP-hard, computation time exponentially grows with the size of the problem (e.g. larger number of feeders, longer planning horizon). It is hence necessary to develop a computationally effective algorithm, namely a GA-based heuristic, to sequence the part-feeding tasks in order to minimize the total traveling time of the mobile robot while satisfying a number of practical constraints. It is also necessary to formulate a mathematical model which allows describing the presented problem. Solutions found by the mathematical model can be used as reference points to quantify the scale of benefits achieved by the GA-based heuristic.

2.4. Mathematical formulation

In this section, an MIP model is developed to determine an optimal sequence in which the mobile robot visits n feeders to process part-feeding tasks. All part-feeding tasks, corresponding to deliveries of SLCs, are known in advance. A method to protect against shortage of parts over a replenishment lead time is also presented to determine the time windows of the part-feeding tasks. This method is based on the (s, Q) inventory system (Silver et al., 1998) where a fixed quantity Q is ordered whenever the inventory position drops to the reorder point s or lower. Hard time window scenario (Toth and Vigo, 2002)

is considered in this case. It means that all part-feeding tasks carried out by the mobile robot can only begin within their time windows. The mobile robot is based at a central warehouse and a limit on the carrying capacity of the mobile robot is imposed. In the following subsections, assumptions, notations, time windows and the formulation of the MIP model are given.

Assumptions

- A mobile robot with a manipulation arm is taken into account in a disturbance free environment.
- The mobile robot can carry one or several SLC(s) at a time.
- All tasks are periodic, independent, and assigned to the same mobile robot.
- Working time and traveling time of the mobile robot between any pair of locations, where either one of the locations could be a feeder or the warehouse, are known.
- The feeding time per part to the machine of a feeder is known and constant.
- All feeders of machines have to be fed up to maximum levels and the mobile robot starts from the warehouse at the initial stage.

Notations

N : set of all tasks ($N = \{0, 1, 2, \ldots, n\}$ where 0: task at the warehouse)

n_i : number of times task i has to be executed

R : set of routes ($R = \{1, 2, \ldots, R_{max}\}$, $R_{max} = \Sigma\, n_i$, $\forall i \in N \setminus \{0\}$)

e_{ik} : release time of subtask k of task i

d_{ik} : due time of subtask k of task i

p_i : periodic time of task i

w_i : working time of mobile robot at location of task i

t_{ij} : traveling time of mobile robot from location of task i to location of task j

c_i : feeding time per part to machine of feeder i

v_i : minimum level of parts in feeder i

u_i : maximum level of parts in feeder i

Q_m : maximum number of SLCs could be carried by mobile robot

T : planning horizon

Decision variables

$$x_{ik}^{jlr} = \begin{cases} 1 & \text{if robot travels from location of task } i \text{ with subtask } k \text{ to location of task } j \text{ with subtask } l \text{ in route } r \\ 0 & \text{otherwise} \end{cases}$$

y_{ik} : route index to which subtask k of task i belongs

s_{ik} : starting time of subtask k of task i

Time windows

The (s, Q) inventory policy is used with the defined characteristics of the feeders to determine the hard time windows of the part-feeding tasks as shown in Figure 2.2, Equation (1), (2), and (3) below.

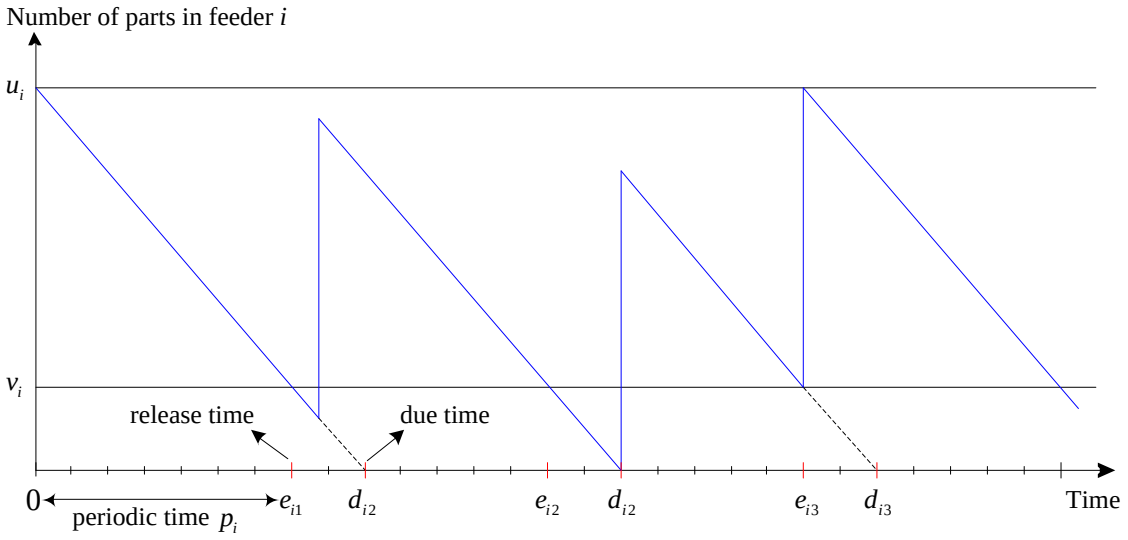

Figure 2.2: Time windows of part-feeding tasks based on the (s, Q) inventory system

$$p_i = (u_i - v_i)c_i, \forall i \in N \setminus \{0\} \quad (1)$$

$$e_{ik} = e_{ik\text{-}1} + p_i, \forall i \in N \setminus \{0\}, k \in \{1, 2, ..., n_i\}, e_{i0} = 0 \quad (2)$$

$$d_{ik} = e_{ik} + (v_i - 0)c_i, \forall i \in N \setminus \{0\}, k \in \{1, 2, ..., n_i\} \quad (3)$$

Because of the periodic characteristics of the tasks for feeder i whose periodic time is calculated as in Equation (1), a number of subtasks must be carried out. The number of subtasks of task i is defined as: $n_i = \lfloor T / p_i \rfloor$. The mobile robot must start processing a subtask k of task i within the associated hard time window of that subtask. It means that the mobile robot is not allowed to arrive at feeder i after the upper bound of the time window. If the mobile robot arrives at feeder i before the lower bound of the time window, it will wait to begin service. The lower bound of the time window, or release time of subtask k of task i, is set to the time when the number of parts inside

feeder i drops to a certain level v_i (Equation (2)). The upper bound of the time window, or due time of subtask k of task i, is defined to the time when there are no parts in feeder i (Equation (3)).

Mixed-integer programming model

The formulation of the MIP model for the presented problem is described as follows. The objective function (4) minimizes the total traveling time of the mobile robot. Constraint (5) ensures that the starting time of any subtask of a task satisfies the time window of that subtask. Constraints (6) and (7) ensure that the mobile robot starts from the warehouse at the initial stage. Constraint (8) eliminates the sub-tours among subtasks of tasks, where Z is a subset of Z_T, where Z_T is a set of all subtasks of tasks at feeders and the warehouse. Constraints (9) and (10) ensure that a subtask of a task is completed exactly once. Constraint (11) forbids the mobile robot to load a higher number of SLCs than its maximum capacity in the number of SLC Q_m. Constraint (12) handles the traveling time requirements between any pair of subtasks of tasks, where L is a given sufficiently large constant. In case two subtasks of the same task or different tasks are connected but are not in the same route, the mobile robot should visit the warehouse to unload empty SLCs and load filled ones. Constraint (13) assigns a subtask of a task to a route and constraint (14) guarantees an ascending sequence of route indices for subtasks of tasks. Constraints (15) and (16) imply the types of variables. In the following, the MIP model is given.

Objective function: $\min \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} t_{ij} x_{ik}^{jlr}$ (4)

Subject to:

$$e_{ik} \le s_{ik} \le d_{ik} \qquad \forall i \in N\setminus\{0\}, k \in \{1,2,...,n_i\} \tag{5}$$

$$\sum_{j\in N\setminus\{0\}}\sum_{l=1}^{n_j} x_{01}^{jl1} = 1 \tag{6}$$

$$\sum_{j\in N\setminus\{0\}}\sum_{l=1}^{n_j}\sum_{r\in R} x_{01}^{jlr} \le 1 \tag{7}$$

$$\sum_{(i,k),(j,l)\in Z} x_{ik}^{jlr} \le |Z| - 1 \quad \forall r \in R, \forall Z \subseteq Z_T, Z_T = \{(i,k) \mid i \in N\setminus\{0\}, k \in \{1,2,...,n_i\}\} \tag{8}$$

$$\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} x_{ik}^{jlr} = 1 \qquad \forall i \in N\setminus\{0\}, k \in \{1,2,...,n_i\} \tag{9}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} x_{ik}^{jlr} = 1 \qquad \forall j \in N \setminus \{0\}, l \in \{1,2,...,n_j\} \tag{10}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N\setminus\{0\}}\sum_{l=1}^{n_j} x_{ik}^{jlr} \le Q_m \qquad \forall r \in R \tag{11}$$

$$s_{ik} + \left(w_i + t_{ij}\sum_{r\in R} x_{ik}^{jlr} \right) - L\left(1 - \sum_{r\in R} x_{ik}^{jlr}\right) + \left(y_{jl} - y_{ik}\right) \times \left(t_{i0} + w_0 + t_{0j} - t_{ij}\right) \le s_{jl} \tag{12}$$

$$\forall i, j \in N, k \in \{1,2,...,n_i\}, l \in \{1,2,...,n_j\}$$

$$y_{jl} = \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} r \times x_{ik}^{jlr} \qquad \forall j \in N \setminus \{0\}, l \in \{1,2,...,n_j\} \tag{13}$$

$$y_{jl} \ge y_{ik}\sum_{r\in R} x_{ik}^{jlr} \qquad \forall i, j \in N, k \in \{1,2,...,n_i\}, l \in \{1,2,...,n_j\} \tag{14}$$

$$x_{ik}^{jlr} \in \{0,1\} \qquad \forall r \in R, \forall i, j \in N, k \in \{1,2,...,n_i\}, l \in \{1,2,...,n_j\} \tag{15}$$

y_{ik}: positive integer variable $\qquad \forall i \in N \setminus \{0\}, k \in \{1,2,...,n_i\}$ (16)

The MIP model contains a number of decision variables that are constrained to have only integer values; for instance, in the presented problem the decision variable x_{ik}^{jlr} to determine the sequence of tasks is equal to either 0 or 1. Integer variables make optimization problems non-convex and thus far more difficult to solve (Gormley and Eisner, 2013). Computer memory and computational time may rise exponentially as the size of problem increases with more added integer variables. Dang et al. (2013b) have shown that the MIP model found the optimal solution within about 5 seconds for a problem instance with 6 subtasks of part-feeding tasks and 864 integer variables, but about 6 hours for another problem instance with 10 subtasks of part-feeding tasks and 4000 integer variables. Hence, in practice the MIP model could be applicable only to small-scale problems, e.g. a real case of one production line with 4 feeders which has been tested at a factory of a Danish company (Hvilshøj et al., 2012b). In other words, the MIP model may be limited to apply to large-scale problems where the mobile robot can serve more production lines, i.e. more feeders, and near-optimal solutions should be achieved within a reasonable computational time. Thus, it is necessary to use another class of methods to deal with the large-scale problems. Although some of the methods are presented in the literature review, GA seems to be a proper and efficient approach to solve the presented problem in the large-scale cases. Therefore, a heuristic based on the GA will be developed in the next section.

2.5. Genetic-algorithm based heuristic

In this section, a genetic algorithm is employed to develop a heuristic. The developed heuristic allows converting the presented problem so that near-optimal solutions can be found. The genetic algorithm-based heuristic as shown in Figure 2.3 is composed of the following main steps: genetic representation and initialization; constraints handling and fitness evaluation; genetic operators consisting of selection, crossover, and mutation; termination criteria.

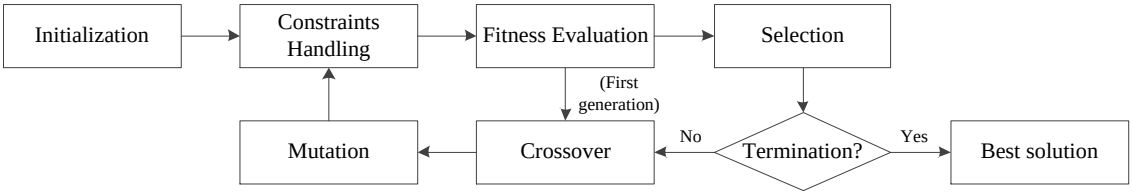

Figure 2.3: Flowchart of genetic algorithm-based heuristic

Genetic presentation and initialization

For the problem under consideration, the natural path representation (Potvin, 1996) is used to represent a chromosome or a solution, which represents an ordering of subtasks of tasks of the mobile robot as shown in Figure 2.4. Each gene in the chromosome consists of two factors. The first factor refers to a task while the second factor implies a subtask of that task. The original length of the chromosome is equal to the total number of subtasks of part-feeding tasks added to the first subtask of warehouse task ($1 + \Sigma n_i$).

0,0	1,1	2,1	1,2	3,1	...	*i,k*	...	*j,l*

Figure 2.4: Genetic representation

For the initial generation, the first factors of genes on a chromosome are randomly filled with tasks at feeders. The frequency of a task is the number of times which that task has to be executed. The second factors of genes having the same first factor/same task are filled in ascending order of subtasks of that task.

Constraints handling and fitness evaluation

After initialization or crossover and mutation operations, chromosomes are adjusted to be valid and their fitness values are calculated. A valid chromosome should satisfy the two constraints: time windows of subtasks of tasks and limit on the carrying capacity Q_m of the mobile robot. For the first constraints, starting time of a subtask of a task

should be in-between release time and due time of that subtask. The second constraint requires that the mobile robot does not serve more subtasks than the number of SLCs it is carrying. A method of handling these constraints is developed and applied to each chromosome in the initial and descendant generations as shown in Figure 2.5 along with the description below:

Step 1: Rearrange genes possessing the same first factor (task) in ascending order of their second factor (subtask). Note that, Step 1 is not employed in the initial generation due to the outcome of the initialization procedure.

Step 2: Temporarily rearrange genes considering only time windows.

Step 2.1: Compute starting time and check the satisfaction of time window constraints for gene (j, l) at position p preceded by gene (i, k) at position $p - 1$.

$s_{jl} = s_{ik} + w_i + t_{ij} \times 1_{\{i \neq j\}}$

If $s_{jl} \leq d_{jl}$ then $s_{jl} = s_{ik} + w_i + t_{ij} \times 1_{\{i \neq j\}}$ or $s_{jl} = e_{jl} \times 1_{\{sjl \leq ejl\}}$, and check if p is the last position on the chromosome. If not, then go to Step 2.6. If so, then go to Step 3.

Otherwise, go to Step 2.2.

Step 2.2: Considering genes from position 1 to the position p on the chromosome, make a set A of positions of genes whose due times are greater than that of gene (j, l).

Step 2.3: Insert gene (j, l) to a position p_a randomly selected from set A.

Step 2.4: Re-compute starting times and check the satisfaction of the time windows constraints for genes from the position p_a to the position p.

If starting time of any gene from the position p_a to the position p does not satisfy its time window, then check if all positions in set A are selected. If not, then go back to Step 2.3. If so, then go to Step 2.5.

Otherwise, check if p is the last position on the chromosome. If not, then go to Step 2.6. If so, then go to Step 3.

Step 2.5: Discard the chromosome, generate a new one instead and go back to Step 1.

Step 2.6: Move to the next gene at position $p + 1$ on the chromosome ($p \leftarrow p + 1$) and go back to Step 2.1.

Step 3: Rearrange genes considering limit on carrying capacity of the robot and time windows.

Step 3.1: Assign the maximum number of SLCs Q_m to the actual number of SLCs, denoted as Q, carried by the robot in a route ($Q \leftarrow Q_m$).

Step 3.2: Considering gene (0, k'), which is a subtask of a warehouse task, at a position q on the chromosome, insert gene (0, k' + 1) at position $q + Q + 1$. Note that, the chromosome length is increased by one unit after each insertion.

Step 3.3: Re-compute starting times and check the satisfaction of the time windows constraints for next Q_m genes.

If starting time of any gene among the next Q_m genes does not satisfy its time window, then go to Step 3.4.

Otherwise, check if (length of chromosome – ($q + Q + 1$)) ≤ Q_m. If not, then go back to Step 3.1. If so, then go to Step 3.8.

Step 3.4: Considering genes from position 1 to position p' of a gene (j', l') which does not satisfy its time window among the next Q_m genes, make a set B of positions of genes whose due times are greater than that of gene (j', l').

Step 3.5: Swap gene (j', l') with a gene at a position p_b randomly selected from set B.

Step 3.6: Re-compute starting times and check the satisfaction of time windows constraints for genes from the position p_b to the position p'.

If the starting time of any gene from the position p_b to the position p' does not satisfy its time window, then check if all positions in set B are selected. If not, then go back to Step 3.5. If so, then go to Step 3.7.

Otherwise, go to Step 3.1.

Step 3.7: Decrease Q by one unit ($Q \leftarrow Q - 1$), and check if Q is 0. If not, then go back to Step 3.2. If so, go back to Step 2.5.

Step 3.8: Insert gene (0, k^*) representing the final subtask of a warehouse task at the end of the chromosome.

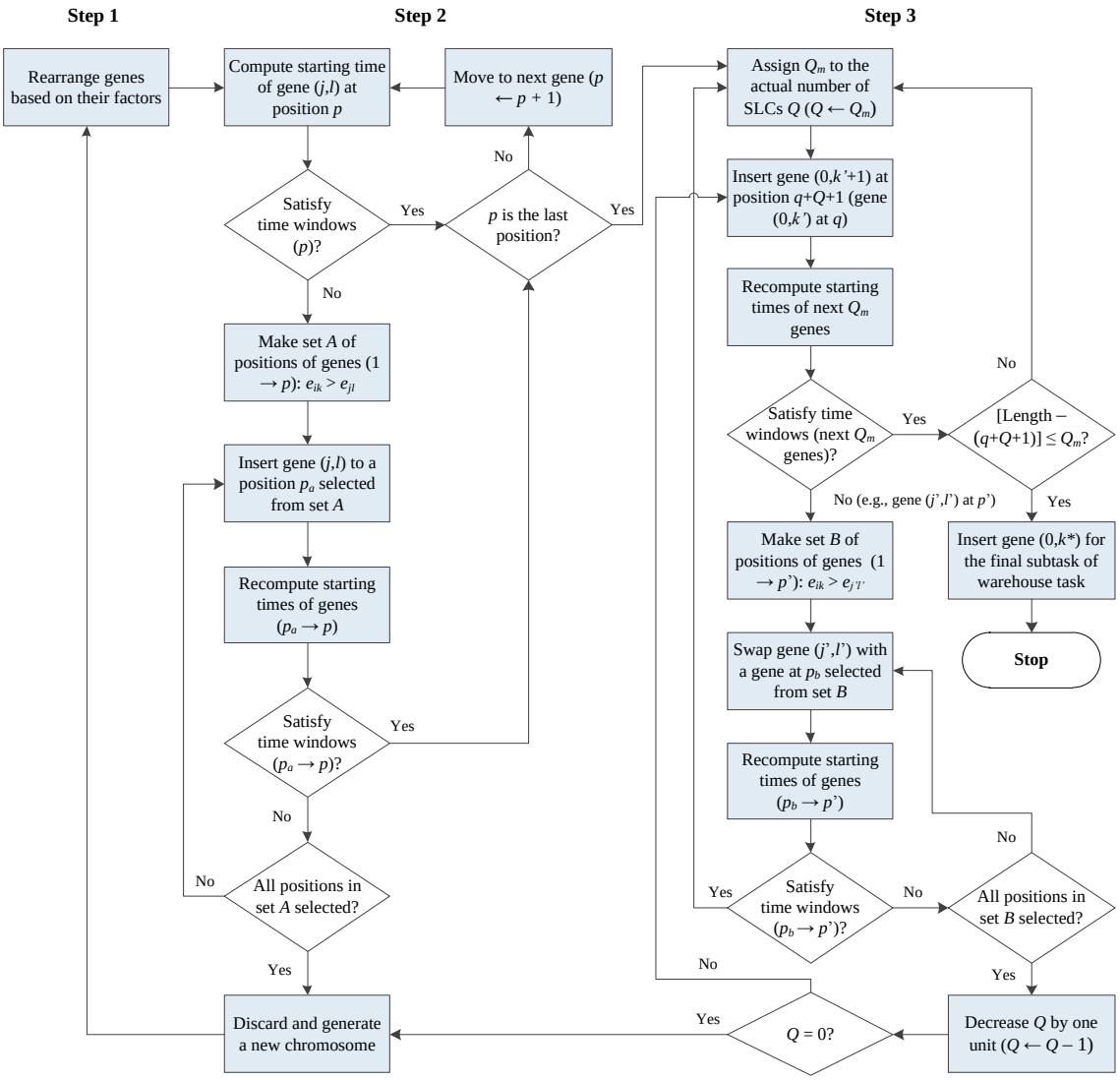

Figure 2.5: Flowchart of handling constraints

Following the method of handling constraints, the fitness evaluation will take place. The fitness value of a chromosome is equal to the total traveling time of the mobile robot, Σt_{ij}, where i, j are the first factors of genes on the chromosome.

Genetic operators

Selection, crossover and mutation are three main genetic operators. For selection, various evolutionary methods can be applied to this problem. (μ + λ) selection is used to choose chromosomes for reproduction. Under this method, μ parents and λ offspring compete for survival and the μ best out of the set of offspring and old parents, i.e. the μ lowest in term of the total traveling time, are selected as the parents of the next generation. This selection mechanism guarantees that the best solutions up to now are always in the parent generation (Dang et al., 2013b).

Crossover operator generates offspring by combining the information contained in the parent chromosomes so that the offspring will have the desirable features from their parents. The Roulette-wheel selection is used in the algorithm, which selects probabilistically the parent chromosomes based on their fitness values (Ho and Ji, 2004; Moon et al., 2006). Then, different crossover operators can be used on the selected parent chromosomes. These are represented by the path presentation, e.g. partially-mapped crossover (PMX), cycle crossover (CX), order crossover (OX), order-based crossover (OBX), and position-based crossover (PBX) (Tsujimura and Gen, 1999; Lin et al., 2006). Although the crossover operators may affect the efficiency of the search process, the quality of solutions is often reasonably close. In the experiment, OX will be used to generate an offspring as described below. Genes which represent subtasks of the warehouse task are removed before two cut points are randomly chosen on the parent chromosomes. The string between these cut points in one of the parents is first copied to the offspring. The remaining positions are then filled by considering the sequence of genes in the other parent starting after the second cut point. When reaching to the end of the offspring, the sequence continues at position 1. The OX operates with probability P_c. Figure 2.6 illustrates the OX procedure.

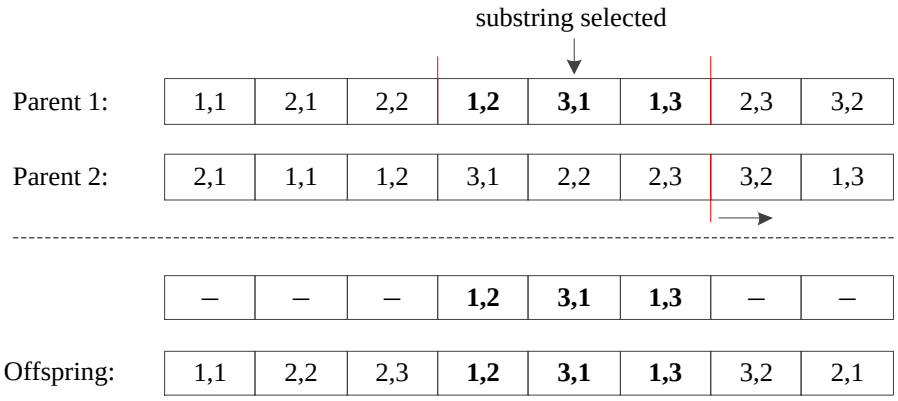

Figure 2.6: Example of order crossover procedure

Whenever an offspring is produced, it undergoes a mutation operator which is applied with probability P_m. The mutation selects two genes within the offspring at random and then swaps these genes to produce heterogeneous chromosomes. This procedure avoids premature convergence of the GA-based heuristic. Note that offspring produced after crossover and mutation operations might not be valid. Therefore, they have to be adjusted by using the method of handling constraints described above.

Termination criteria

Termination criteria are employed to determine when the GA-based heuristic should be stopped. On the one hand, computation time plays an important role in practice because taking decisions on which sequences the robot should serve feeders is a part of the real-time operations of production planners. In other words, the best solution is required to be quickly obtained. Therefore, if the best solutions over generations do not converge, the maximum computation time (CT_m) would be used to stop the run. On the other hand, if the best solution does not improve over a number of consecutive generations (G_c), it would not be valuable to continue searching. The up-to-date best solution is then returned as the near-optimal solution. However, it should be noted that high-quality local optima might exist (in case of existing feasible solutions) due to the combinatorial nature of the problem (Dang et al., 2013b).

2.6. Numerical examples and comparisons

To examine the performance of the MIP model and GA-based heuristic, a case study including two different demonstrations has been investigated with real data of Grundfos A/S, a Danish company which is one of the world's leading pump manufacturers. A part of Grundfos production facilities, CR (compression ratio) 1-2-3 impeller line at CR factory, has been used to implement these demonstrations. An extension of the case study considering several reasonable assumptions has been also conducted to make the evaluation of both approaches more convincing. Finally, various problem instances are randomly generated and tested in order to provide more persuasive evidence of the performance of the proposed heuristic. The MIP model has been coded and solved by the mathematical modelling language ILOG CPLEX, while the proposed heuristic has been programmed in VB.NET. All the experiment run on a PC having an Intel® Core i5 2.67 GHz processor and 4 GB RAM.

Case Study

The chosen area for the case study is the CR 1-2-3 impeller production line shown in Figure 2.8 that manufactures impellers for the CR pumps. The CR line consists of a warehouse and four feeders that have to be served by the mobile robot. The warehouse is indexed 0 and the feeders are indexed from 1 to 4 (N = {0, 1, 2, 3, 4}) and named Back Plate, Van Feeder 1, Van Feeder 2, and Front Plate respectively. Furthermore,

different feeders are filled by different types of parts, namely back plates for feeder 1, vanes for feeder 2 and 3 and front plates for feeder 4. To produce an impeller on the CR 1-2-3 line, these three types of parts are automatically assembled. The impeller consists of six vanes with three from feeder 2 and the other three from feeder 3, one front plate from feeder 4 and one back plate from feeder 1. Figure 2.7 shows the different parts of an impeller, while Figure 2.8 particularly illustrates the production area where the case study has been implemented. The safety fences and warning signs are used as depicted in Figure 2.8 to ensure that no people enter the area as well as to prevent the mobile robot leaving that area while the demonstrations are taking place.

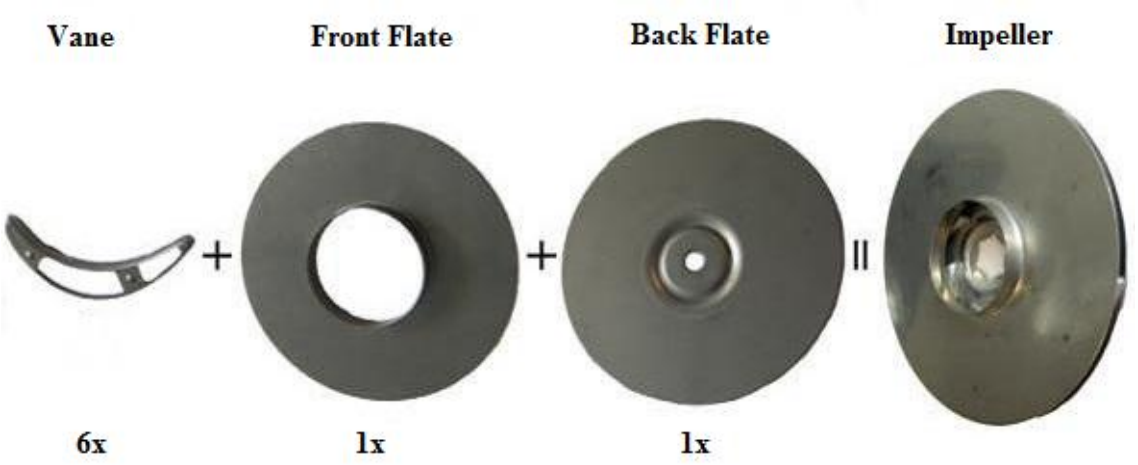

Figure 2.7: Different parts of an impeller produced on the CR 1-2-3 line

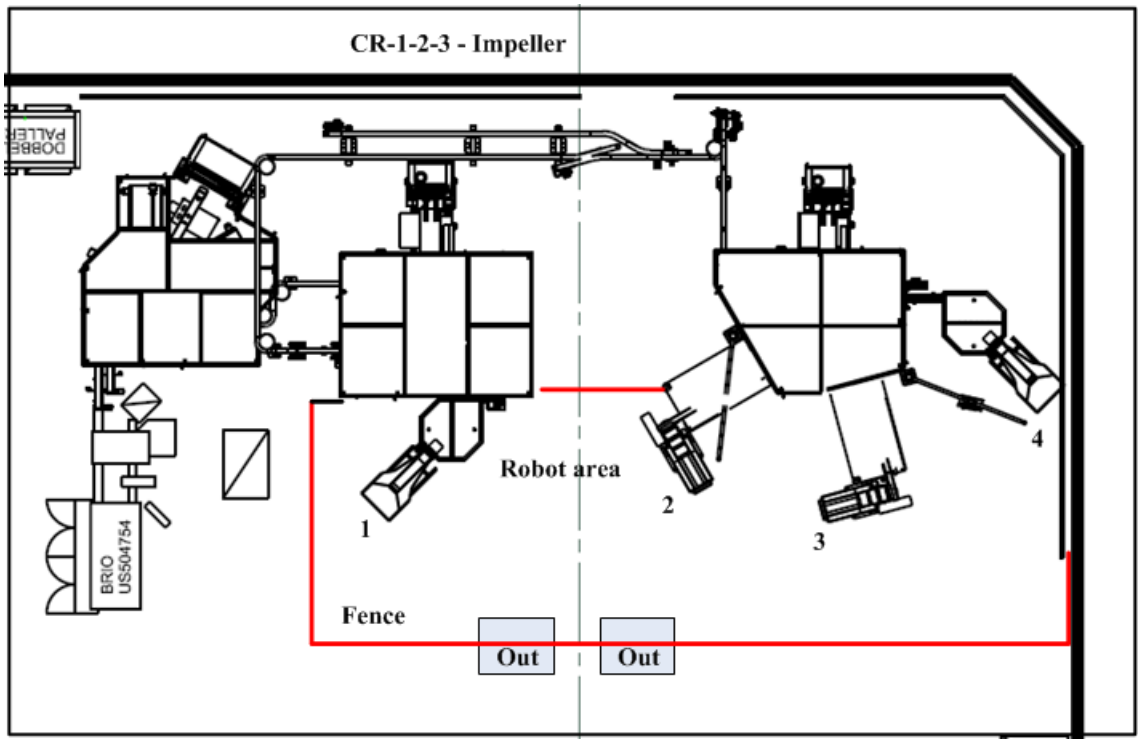

Figure 2.8: CR 1-2-3 impeller production line

The following data is taken from the Manufacturing Execution System (a computerized system located between the ERP and process control system, and used in management of entire manufacturing processes including equipment, personnel, and support services) as well as real tests on the shop floor of the CR factory. This data is used as input for two demonstrations of the case study. The average number of parts per SLC fed to feeder 1 or 4 is 125 (approximately 2 kg/SLC), while the average number of parts per SLC fed to feeder 2 and 3 is 1100 (approximately 1 kg/SLC). The maximum levels, minimum levels, and feeding time/part to machines of feeders are given in Table 2.1. The feeding time/part to machines of feeders are derived from the cycle time of the CR 1-2-3 line of 4.5 seconds (Reuther et al., 2010). Specifically feeders 1 and 4 feed machines with one back plate and one front plate every 4.5 seconds while feeders 2 and 3 feed machines with one vane every 1.5 seconds (3 vanes for every 4.5 seconds). The working times of the robot at the feeders are given in Table 2.2 and Table 2.3 shows the travelling times of the robot from locations to locations (feeder 0 means the warehouse).

Table 2.1: Maximum levels, minimum levels, and feeding time per part to machines of feeders

Feeder/Task	1	2	3	4
Maximum level (part)	250	2000	2000	250
Minimum level (part)	125	900	900	125
Feeding time/part (second/part)	4.5	1.5	1.5	4.5

Table 2.2: Working times of robot at locations (seconds)

Feeder/Task	0	1	2	3	4
Working time of robot	90	42	42	42	42

Table 2.3: Traveling times of robot from locations to others (seconds)

From feeder \ To feeder	0	1	2	3	4
0	0	34	37	34	40
1	39	0	17	34	50
2	35	17	0	35	49
3	34	33	35	0	47
4	36	47	48	46	0

In the initial design, the mobile robot has the capability to carry up to three SLCs at a time while performing part-feeding tasks at the feeders. Hence, two different demonstrations of the case study have been investigated corresponding to the two maximum numbers of SLCs, $Q_m = 2$ and $Q_m = 3$, and with the planning horizon T of approximately 45 minutes due to the battery limit of the robot. The parameters p_i, e_{ik},

and d_{ik} are respectively calculated based on Equation (1), (2), and (3) with the data in Table 2.3. As a result, the total number of subtask of part-feeding tasks is 10 and the number of decision variables is 4040 in each demonstration of the case study. For GA parameters, Dang et al. (2013b) carried out statistical analyses to examine the effect of these parameters on the efficiency of the proposed heuristic and also to set the values for these parameters. By performing the statistical analyses, Dang et al. (2013b) proved that population size (N_p), P_c, P_m, G_c, and CT_m affect the objective values of the proposed heuristic and set N_p, P_c, P_m, G_c, and CT_m to be 100, 0.8, 0.1, 100, and 60, respectively.

In each demonstration of the case study, two cases of the MIP are investigated for comparing the performances with the GA-based heuristic. The first case is carried out when giving the same maximum computation time CT_m (60 seconds) as the GA-based heuristic while the second case is performed without limit on the computation/run time. Table 2.4 gives solutions of the MIP and proposed heuristic on the objective value and computation time (in seconds) for the demonstrations at the CR factory. The cells containing a "–" symbol indicate that the results of the corresponding problems cannot be obtained by using the corresponding approach. From Table 2.4, it can be seen that when giving the same maximum computation time CT_m as the GA-based heuristic, the MIP is not able to find any feasible solution. In contrast the proposed heuristic found solutions for the demonstrations (objective values of 504 seconds/8.4 minutes in D-1 and 396 seconds/6.6 minutes in D-2). These objective values found through the heuristic are greater than those found by the MIP when the run time of the MIP is unlimited. However, the differences are only about 3% and this is deemed to be an acceptable error. Furthermore, the computation time shows that use of the MIP is too time-consuming whereas the proposed heuristic significantly faster obtains near-optimal solutions (approximately 6 hours in D-1 or 2.3 hours in D-2 as opposed to less than a second). It also reveals that the higher maximum numbers of SLCs the robot can carry, the less it has to travel around the manufacturing cell (8.4 minutes with Q_m of 2 as opposed to 6.6 minutes with Q_m of 3). Sequences of part-feeding tasks found by the heuristic are depicted using Gantt charts in Figure 2.9.

Table 2.4: Solutions of the case study under MIP and GA-based heuristic

Demo	Q_m	MIP (limited to CT_m)		MIP (unlimited)		GA-based heuristic		Penalty of the heuristic (%) vs. MIP (unlimited)
		Objective value (s)	Computation time (s)	Objective value (s)	Computation time (s)	Objective value (s)	Computation time (s)	
D-1	2	–	–	488	21589.34	504	<1	3.28
D-2	3	–	–	384	8377.27	396	<1	3.13

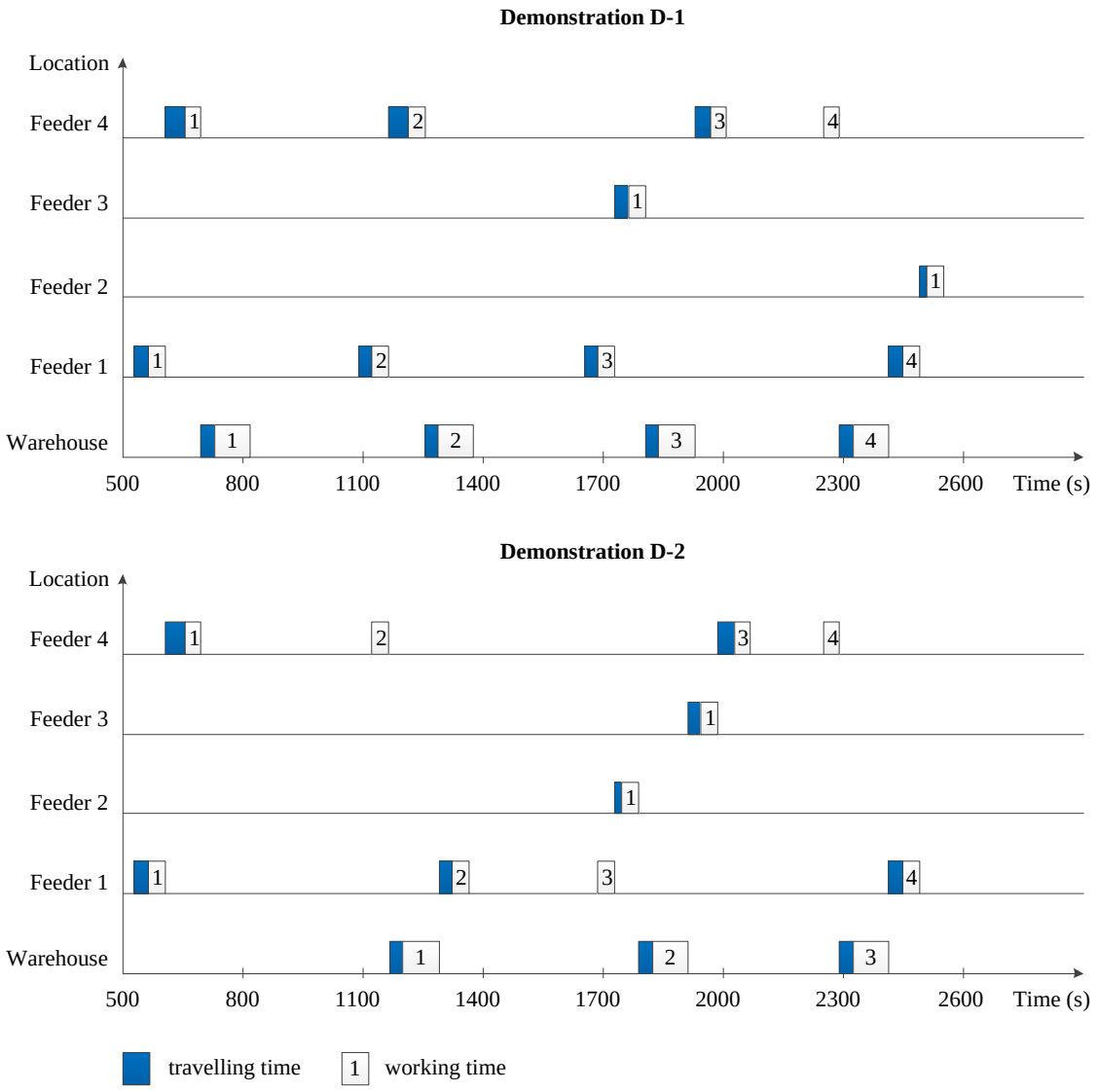

Figure 2.9: Gantt charts for the solutions of two demonstrations of the case study

To make the evaluation more convincing, the case study is extended by assuming that the robot has the capability of carrying up to 4 SLCs at a time and the battery limit of the robot allows it to work up to 8 hours (a full production shift). Further comparisons of the objective value and computation time for the MIP and GA-based heuristic are presented in Table 2.5. Note that in this extended experiment, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs. The total number subtask of all tasks and number of decision variables are also given. From Table 2.5, it can be seen that the proposed heuristic has the capability of solving larger problems while the MIP cannot find any feasible solution for problems of this scale. It also shows that in case of full production shift of 8 hours, the proposed heuristic is able to find the best solutions in less than 30 seconds.

Moreover, the standard deviation of the objective value is quite small in comparison with the average. The GA-based heuristic, therefore, demonstrates efficiency in solving the larger problems.

Table 2.5: Comparison between MIP and GA-based heuristic for extension of the case study

Q_m	T (hour)	Total subtasks of tasks	Number of variables	MIP		GA-based heuristic			
				Objective value (s)	Computation time (s)	Objective value (s)		Computation time (s)	
						Average	Std. Dev.	Average	Std. Dev.
2	1	16	16448	–	–	827	0	0.35	0.04
	2	32	131200	–	–	1596	2	1.16	0.10
	4	66	1150248	–	–	3205	39	5.01	1.37
	8	136	10062368	–	–	6738	70	25.09	7.28
3	1	16	16448	–	–	648	7	0.34	0.06
	2	32	131200	–	–	1231	17	1.32	0.27
	4	66	1150248	–	–	2615	44	4.96	0.99
	8	136	10062368	–	–	5667	83	16.46	3.98
4	1	16	16448	–	–	628	4	0.31	0.06
	2	32	131200	–	–	1206	6	1.05	0.15
	4	66	1150248	–	–	2497	30	4.24	1.21
	8	136	10062368	–	–	5223	85	15.49	2.36

Computational experiments

In this section, the performance of the proposed heuristic will be tested on a large number of problem instances. 20 problems are generated with different numbers of feeders, maximum numbers of SLCs, planning horizons and other system parameters. The number of feeders and the maximum number of SLCs are randomly generated in the ranges of [3, 20] and [2, 4], respectively. The planning horizons in hours are 1, 2, 4, and 8 (corresponding to an eighth, a quarter, half, and full of the production shift). The maximum and minimum levels of parts in feeders are respectively uniformly distributed within the ranges of [500, 2000] and [100, 1000] while feeding time/part to machines of feeders (in seconds) are generated in the interval [1.5, 6.5]. The working times of the robot in seconds at feeders and the warehouse are respectively distributed within the range of [40, 60] and [80, 100] while the traveling times of the robot in seconds are generated in the interval [20, 60]. Note that the time/cost matrices of the generated traveling times should satisfy the triangle inequality. The comparisons between the MIP and GA-based heuristic for 20 randomly generated problems are presented in Table 2.6.

Similar to the extension of the case study, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and the computation times of the proposed heuristic are the average of 30 runs. General information for these 20 problems is also shown in Table 2.6.

Table 2.6: Comparison between MIP and GA-based heuristic for 20 randomly generated problems

No.	No. of feeders	Q_m	T (hour)	Total subtasks of tasks	Number of variables	MIP		GA-based heuristic			
						Objective value (s)	Computation time (s)	Objective value (s)		Computation time (s)	
								Average	Std. Dev.	Average	Std. Dev.
1	7	3	1	10	4040	790	60.00	377	9	0.22	0.04
2	15	2	1	25	62600	–	–	986	14	1.52	0.19
3	6	2	1	15	13560	–	–	775	10	0.29	0.03
4	3	4	1	6	888	184	4.93	184	0	0.14	0.08
5	16	3	1	36	186768	–	–	1120	22	2.61	0.39
6	5	2	2	28	87920	–	–	1287	7	0.76	0.15
7	11	3	2	53	595720	–	–	1657	33	4.00	0.84
8	10	4	2	44	340912	–	–	1006	31	2.78	0.50
9	4	3	2	13	8840	811	60.00	428	0	0.18	0.01
10	18	4	2	70	1372280	–	–	2156	47	39.72	5.34
11	15	3	4	106	4764488	–	–	3504	76	28.83	6.07
12	12	4	4	99	3881592	–	–	2998	59	14.61	1.98
13	9	4	4	84	2371152	–	–	2229	38	5.97	0.91
14	19	3	4	115	6083960	–	–	4732	33	60.00	0.00
15	11	2	4	92	3115120	–	–	4070	28	30.73	3.58
16	8	3	8	113	5772040	–	–	4498	79	13.18	2.29
17	13	2	8	154	14609672	–	–	6696	71	42.89	7.47
18	17	4	8	200	32000800	–	–	7342	95	60.00	0.00
19	14	3	8	170	19652680	–	–	6249	83	48.75	5.08
20	20	4	8	220	42592880	–	–	8071	99	60.00	0.00

It can be observed from Table 2.6 that the GA-based heuristic is superior to the MIP for large problems. The MIP found feasible solutions for problem instances 1, 4, and 9. However, the solutions found by the MIP are much worse than those found by the proposed heuristic (except problem instance 4 in which both approaches found the optimal solution). In addition, the MIP cannot find any feasible solution for the other problems. The GA-based heuristic, by contrast, is able to find best solutions for all 20 problem instances. As the size of the problem increases especially the total subtasks of all tasks, the computation time of the proposed heuristic becomes longer. The results also show that with the same planning horizon, the same or nearly the same number of

feeders, the less maximum number of SLCs the robot can carry, the more difficult it is for the proposed heuristic to generate feasible sequences of part-feeding tasks, thus the longer computation time it may require to obtain best solutions (e.g. problem instances 12 and 15). Furthermore, in terms of the objective value, the standard deviation is quite small in comparison with the average. These results provide more persuasive evidence to prove that the GA-based heuristic performs effectively.

2.7. Conclusions

This chapter presents the results of a study of the novel problem of scheduling a single mobile robot in order to perform part-feeding tasks on production lines. The robot has been assigned to the production by applying the bartender concept. To accomplish all tasks in a planning horizon within the allowable limit of battery capacity and power, it is important for production planners to determine feeding sequences which minimize the total travelling time of the robot. This must be done while taking into account a number of practical constraints. The main novelty of this research lies in the simultaneous consideration of hard time windows of part-feeding tasks and multiple delivery routes in case of the single mobile robot. An MIP model to find exact optimal solutions for the problem is developed. Due to the NP-hard nature of the problem, this solution approach is only applicable to small-scale problems with few feeders and short planning horizon. To deal with large-scale applications, a GA-based heuristic is then proposed to find near-optimal solutions. The quality of these solutions could then be evaluated by using the MIP solutions as reference points to quantify the scale of benefits. The results of the real case study shows that use of the MIP is too time-consuming whereas the proposed heuristic is significantly faster in finding near-optimal solutions. Further experiments provide persuasive evidence that the proposed heuristic is capable of solving problems of various sizes and more efficient than the MIP in terms of the objective value when giving the same maximum computation time. It can be also observed that the larger number of SLCs the mobile robot can carry, the easier the proposed heuristic can generate feasible sequences and the less computation time may be required to obtain the best solution.

So far, this chapter has only taken into account a single mobile robot with the capability of transporting material. In fact, a fleet of mobile robots working together in a manufacturing system can also be considered. In addition to transporting, these mobile

robots possess the capability of carrying out more advanced tasks, e.g. machine tending, pre-assembly, and quality inspection at different machines, workstations, or production lines. Furthermore, mobile robots are flexible enough to switch from one type of tasks, e.g. transporting tasks to another type of tasks, e.g. pre-assembly tasks. Therefore, it is possible to extend the research by considering multiple mobile robots which help to serve and satisfy more production needs. In the next chapter, this case will be taken into account.

3. Multiple mobile robots with preemptive tasks

3.1. Introduction

Despite the fact that in Chapter 2 a mobile robot scheduling problem has been solved, only one mobile robot and one type of tasks, i.e. part-feeding, have been investigated. The major advantage of mobile robots is that it is possible to configure them for various industrial tasks. This means mobile robots can have enough flexibility to switch from one type of tasks to another. Furthermore, a fleet of mobile robots can be considered, which helps to serve and satisfy more production requirements. Therefore, within the scope of this chapter, a problem of simultaneous scheduling of machines and multiple mobile robots in a flexible manufacturing system (FMS) (Buzacott and Yao, 1986) is addressed. The FMS consists of a number of operations of different tasks processed on a set of machines and a set of mobile robots. Some are non-preemptive tasks which require the mobile robots to perform transportation of materials/parts between some machines. Meanwhile, the others are preemptive tasks which are processed by the mobile robots on the other machines. During operation, it is possible to assign the mobile robots to transportation of non-preemptive tasks while these mobile robots are processing preemptive tasks. The performance criterion is to minimize the makespan. As making schedules of machines and mobile robots is part of real-time operations, it is required to quickly find the best schedules. In addition, the complexity of the problem rapidly increases with the number of tasks and mobile robots. Thus in this chapter, a computationally efficient method, namely a GA-based heuristic, is developed to solve the problem. A mixed-integer programming (MIP) model is also formulated to evaluate the performance of the proposed heuristic. In the next section, the related research will be surveyed.

3.2. Survey of literature

The problem of simultaneous scheduling of machines and mobile robots in an FMS has been modeled in several respects comparable to the joint scheduling problems of machines and AGVs. However, it is different from the problems concerning AGVs in the sense that besides transporting tasks, mobile robots are capable of carrying out manufacturing tasks on the shop floor level. Moreover, they are flexible enough to

switch between manufacturing tasks and transporting tasks. Several approaches and models for exact or (meta-)heuristic algorithms have been proposed to address problems of this type. Exact methods are mainly used for the research of simple or particular FMS (Deroussi et al., 2008). Blazewicz et al. (1991) study the model of an FMS considering both machine and vehicle scheduling, and then propose a dynamic programming approach to construct optimal production and vehicle schedules. This FMS is later formulated in MIP by Bilge and Ulusoy (1995). According to the authors, the resulting model is intractable in practice due to its nonlinearity and its size. Caumond et al. (2009) study the linear formulation of an FMS considering the maximum number of jobs, limited input/output buffer capacities, empty-vehicle trips and no-move-ahead trips concurrently. However, only one AGV is taken into account as a special case of the general FMS. For small and medium scheduling problems, the aforementioned techniques can guarantee the optimal solutions. Nevertheless, they are not practical for large and complex problems because the computation time is often too long, e.g. many hours or even many days (Khayat et al., 2006; Gen and Lin, 2008).

Heuristic methods are well adapted to study most of the FMS. On the one hand, some works are dedicated to simplified forms of the material handling system of the FMS considering only one transport device. As illustration, Soylu et al. (2000), Hurink and Knust (2002), Lacomme et al. (2005) propose, respectively, neural network, tabu search, and heuristic branch-and-bound approaches for scheduling of the FMS based on a single AGV or transport robot. On the other hand, many works are undertaken on the FMS scheduling with multiple AGVs. Ulusoy and Bilge (1993) and Bilge and Ulusoy (1995) propose an iterative method based on the decomposition of the master problem into the two sub-problems: machine scheduling and vehicle scheduling. A heuristic algorithm generates machine schedules to solve the first problem. A solution heuristic based on sliding-time-window approach is introduced to find feasible solutions to the vehicle scheduling problem given the machine schedules. Ulusoy et al. (1997) deal with the problem of concurrent scheduling of machines and AGVs by proposing a genetic algorithm which provides a suitable coding scheme to represent both dimensions of the search space: operation sequencing and AGV assignment. Abdelmaguid et al. (2004) introduce a hybrid method which is composed of a GA for scheduling of machines and a heuristic for scheduling of vehicles. Reddy and Rao (2006) present a hybrid multi-objective GA to solve the simultaneous scheduling of machines and AGVs in an FMS in which makespan, flow time, and tardiness are performance criteria. Jerald et al.

(2006) address the problem of simultaneous scheduling of parts and AGVs in an FMS environment using adaptive GA. Lin et al. (2006) model an AGV system by using network structure and propose an effective evolutionary approach for solving a kind of AGV problems. Deroussi et al. (2008) describe an efficient neighboring system which is implemented into three different meta-heuristics and a new solution representation based on vehicles rather than machines. Lacomme et al. (2013) introduce a framework based on a disjunctive graph to modelize the joint scheduling problem and on a memetic algorithm for machines and identical AGVs scheduling.

Although a number of researches related to the class of mobile robot scheduling problems have been done, the problem of simultaneous scheduling of machines and mobile robots with preemptive tasks in an FMS has received little focus in the literature. The considered mobile robots in this problem have the capabilities to not only transport non-preemptive tasks between some machines similar to material handling devices but also processing preemptive tasks on other machines by using their manipulation arms. These mobile robots are allowed to interrupt their processing tasks to do transportation of non-preemptive tasks when needed. These facts constitute the main novelty of the problem. The surveyed approach are not well suited and cannot be directly used to solve this problem due to the lack of a suitable mechanism for scheduling and routing mobile robots in relation to scheduling machines while taking into account preemptive tasks. Thus, it is necessary to develop a proper and efficient method to deal with this problem.

3.3. Problem Description

Flexible manufacturing systems are highly automated production systems capable of producing a variety of part/component types. Such manufacturing system originally includes intelligent and flexible machines, automated storage and retrieval systems, and material handling devices such as AGVs or robots. Furthermore, the development of automation technology has significantly changed the manufacturing equipment on the shop floor. With these changes, mobile robots have been designed and manufactured so that they can combine the flexibility of service robots, e.g. AGVs, with the efficiency of industrial robots, i.e. dedicated and fixed robots in industry. This enables these mobile robots not only to transport parts/components to machines but also to operate machines to process tasks. Thus, they have been widely employed in not only small companies, which have focus on exact applications and a small range of products, but also large

companies, which can diversify their applications in a longer term and larger range. For instance in a pump parts manufacturing system, a mobile robot is assigned to carry out a pre-assembly task, and during that operation it also has to transport and feed material into some production lines when needed. In this chapter, the coordination between mobile robots and machines in an FMS is considered. Figure 3.1 below shows a typical layout of the FMS with mobile robots.

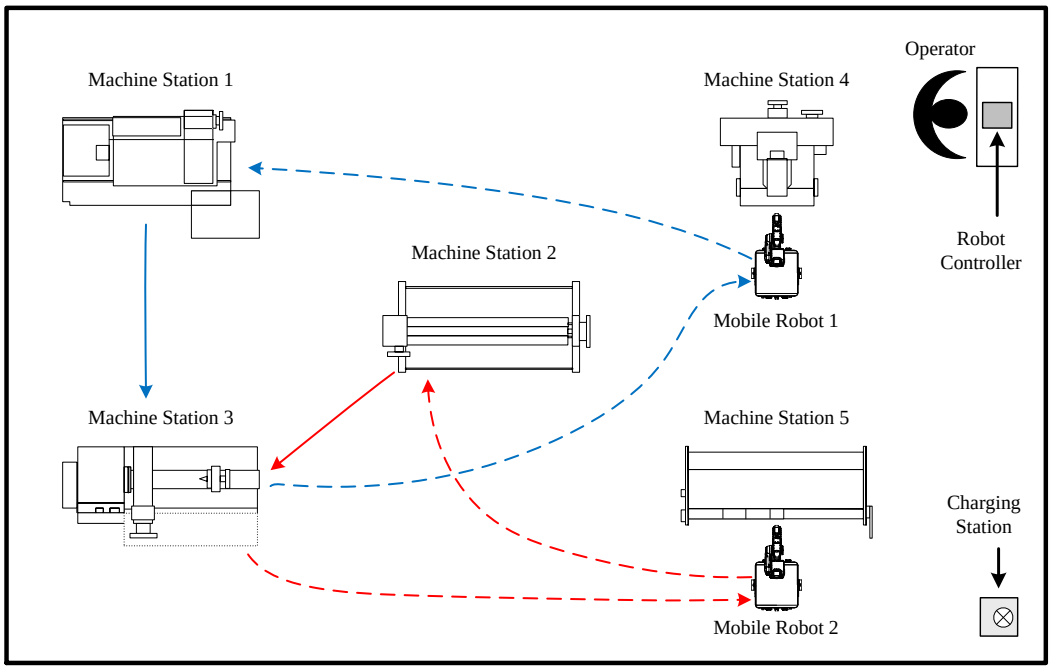

Figure 3.1: Typical layout of the FMS with mobile robots

The FMS consists of a number of tasks processed on a set of machines, and a set of mobile robots. These tasks are classified into two types which are non-preemptive and preemptive. Each non-preemptive task consists of a set of operations that cannot be interrupted, i.e. each operation in this type must be executed without interruption from its starting time to its ending time (Yun, 2002). On the other hand, each preemptive task considered in this chapter has only one operation that can be interrupted at any time to let some other operations execute. There is no restriction on the number of interruptions or on the duration of an interruption. During operation, the non-preemptive tasks require mobile robots to transport materials/parts between some machines while the preemptive tasks need the participation of mobile robots in the processing on the other machines. Each mobile robot may carry out the transportation of different non-preemptive tasks, but it is assigned to process only one preemptive task on a specific machine. As being occupied by a preemptive task, a mobile robot may be invoked for transportation of a non-preemptive task at some points in the scheduling period. This mobile robot will

pause processing of the preemptive task, carry out transportation of the non-preemptive task, and go back to processing the preemptive task if the preemptive task has not been finished. In practice, e.g. in a pump parts manufacturing factory, production operators may set the maximum number of operations of non-preemptive tasks which mobile robots can transport each time being away from their preemptive tasks. This prevents the mobile robots from leaving their preemptive tasks for a long period of time, which may lead to the cancellation of these tasks due to some practical issues on the shop floor. To some extent, this also helps to increase the utilization of these mobile robots. Within the scope of this study, any mobile robot is set to come back to its processing machine after each achieved transportation. An example illustrating such preemption case is given in Figure 3.2. The objective is to find a schedule that minimizes the time required to complete all tasks, i.e. makespan.

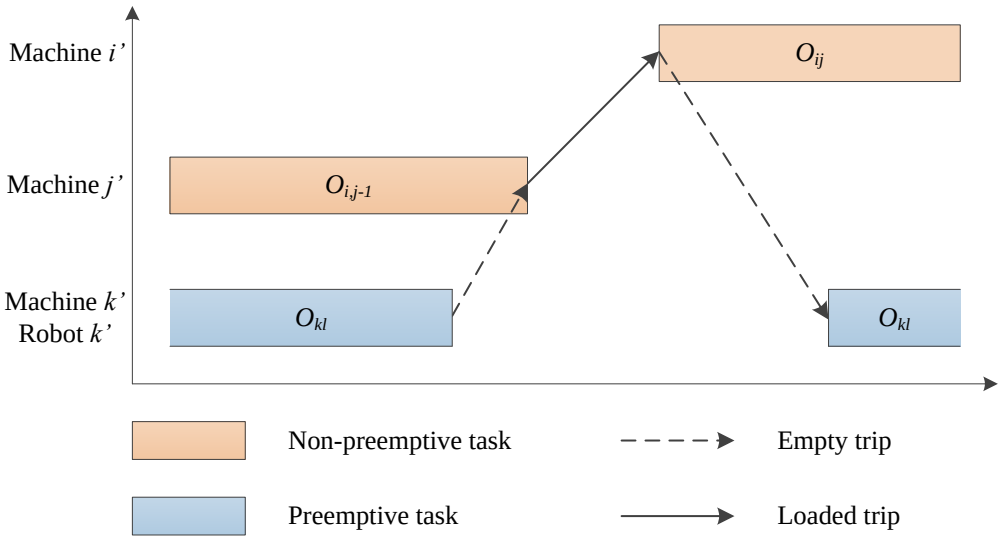

Figure 3.2: Illustration of a preemption case

To enable the construction of a schedule of machines and mobile robots, the following assumptions are made.

- Each task is available at the beginning of the scheduling period.
- The first operation of each task is available at a machine at the beginning of the scheduling period.
- Each operation sequence of each task (the route of each part type) is available before making scheduling decisions.
- Each mobile robot can transport only one kind of parts at a time.
- There is sufficient input and output buffer space at each machine.
- Traveling time is only machine-dependent and deterministic.

- Loading and unloading time are included in the traveling time of loaded trips.
- Processing time is deterministic.

Note that making schedules of machines and mobile robots is part of real-time operations of production planners. It means that the best solutions should be quickly obtained. Moreover, as the problem is NP-hard, the computation time exponentially grows with the size of the problem (e.g. larger number of operations of tasks and mobile robots). It is thus necessary to develop a computationally effective algorithm, namely a GA-based heuristic, to schedule machines and mobile robot in order to minimize the makespan while satisfying a number of practical constraints such as precedence and resource (machines and mobile robots) constraints. It is also necessary to formulate a mathematical model which allows describing the presented problem. Solutions found by the mathematical model can be used as reference points to quantify the scale of benefits achieved by the GA-based heuristic.

3.4. Mathematical formulation

In this section, an MIP model is formulated to determine an optimal schedule for the problem of simultaneous scheduling of machines and mobile robots with preemptive tasks. The notations and formulation of the MIP model are given in the following.

Notations

i, i', i^* : index of tasks

j, j', j^* : index of operations

k : index of mobile robots

N : set of non-preemptive tasks

P : set of preemptive tasks

M : set of machines

R : set of mobile robots

O_m : set of operations of non-preemptive tasks to be performed on machine m

o_{ij} : the j-th operation of task i

T_{ij} : transportation of operation o_{ij}

n_i : number of operations for task i

p_{ij} : processing time of operation o_{ij}

$t_{ij,i'j'}$: traveling time from machine of operation o_{ij} to machine of operation $o_{i'j'}$

Decision variables

T : time required to complete all tasks

s_{ij}^{p} : starting time of operation o_{ij}

s_{ij}^{t} : starting time of transportation T_{ij}

$$y_{iji'j'} = \begin{cases} 1 & \text{if operation } o_{ij} \text{ precedes operation } o_{i'j'} \\ 0 & \text{otherwise} \end{cases}$$

$$z_{iji'j'} = \begin{cases} 1 & \text{if transportation } T_{ij} \text{ precedes transportation } T_{i'j'} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if robot } k \text{ carries out transportation } T_{ij} \text{ for operation } o_{ij} \\ 0 & \text{otherwise} \end{cases}$$

Mixed-integer programming model

The formulation of the MIP model for the presented problem is described as follows. The objective function (1) minimizes the makespan T. Constraints (2) and (3) represent the operation non-overlapping constraints which imply that a machine cannot process more than one operation at a time, where L is a very large number. Constraints (4) and (5) represent the mobile robot non-overlapping constraints which imply that a mobile robot cannot perform more than one transportation task at a time. Constraint (6) ensures that the transportation of an operation could only start after the predecessor of that operation finishes. Constraint (7) ensures that an operation could only start after the transportation of that operation completes. Constraint (8) ensures that the transportation of an operation is carried by only one mobile robot. Constraints (9) and (10) ensure that completion time of non-preemptive operations (operations of non-preemptive tasks) and preemptive operations (operations of preemptive tasks) respectively cannot exceed the makespan T. Constraints (11) and (12) imply the types of variables. The MIP model is given in the following:

Objective function: minimize T (1)

Subject to:

$$s_{ij}^{p} - s_{i'j'}^{p} + Ly_{iji'j'} \geq p_{i'j'} \qquad \forall (i,j),(i',j') \in O_m, m \in M \tag{2}$$

$$s_{i'j'}^{p} - s_{ij}^{p} + L\left(1 - y_{iji'j'}\right) \geq p_{ij} \qquad \forall (i,j),(i',j') \in O_m, m \in M \tag{3}$$

$$s_{i'j'}^{t} - s_{ij}^{t} + L\left(1 - z_{iji'j'}\right) \geq t_{ij-1,ij} + t_{ij,i^*j^*} + t_{i^*j^*,i'j'} \tag{4}$$

$$\forall i, i' \in N,\ j = 2,...,n_i,\ j' = 2,...,n_{i'}, \forall i^* \in P,\ j^* = 1$$

$$z_{iji'j'} + z_{i'j'ij} = x_{ijk} x_{i'j'k} \qquad \forall i, i' \in N,\ j = 2,...,n_i,\ j' = 2,...,n_{i'} \qquad (5)$$

$$s_{ij}^p + p_{ij} \le s_{i,j+1}^t \qquad \forall i \in N,\ j = 1,...,n_i - 1 \qquad (6)$$

$$s_{ij}^t + t_{ij-1,ij} \le s_{ij}^p \qquad \forall i \in N,\ j = 2,...,n_i \qquad (7)$$

$$\sum_{k \in R} x_{ijk} = 1 \qquad \forall i \in N,\ j = 1,...,n_i \qquad (8)$$

$$s_{ij}^p + p_{ij} \le T \qquad \forall i \in N,\ j = 1,...,n_i \qquad (9)$$

$$s_{i^*j^*}^p + p_{i^*j^*} + \sum_{i \in N} \sum_{j=2}^{n_i} x_{ijk} \left(t_{i^*j^*,ij-1} + t_{ij-1,ij} + t_{ij,i^*j^*} \right) \le T \qquad \forall i^* \in P,\ j^* = 1, k \in R \qquad (10)$$

$$x_{ijk}, y_{iji'j'}, z_{iji'j'} \in \{0,1\} \qquad \forall i, i' \in N,\ j = 1,...,n_i,\ j' = 1,...,n_{i'}, k \in R \qquad (11)$$

$$s_{ij}^p, s_{ij}^t \ge 0 \qquad \forall i \in N,\ j = 1,...,n_i \qquad (12)$$

The MIP model contains a number of decision variables that are constrained to have only integer values; for instance in the presented problem the decision variables x_{ijk}, $y_{iji'j'}$, and $z_{iji'j'}$ equal either 0 or 1. Integer variables make optimization problems non-convex and thus far more difficult to solve (Gormley and Eisner, 2013). Computer memory and computation time may rise exponentially as the size of problem increases with more added integer variables. Thus in practice the MIP model could be applicable only to small-scale problems. In other words, the MIP model may be limited to apply to large-scale problems where the mobile robots can carry out more tasks. Therefore, it is necessary to use another class of methods to solve the large-scale problems. Among some of the methods presented in the survey of literature, GA seems to be a proper and efficient approach to deal with the presented problem in the large-scale cases. Hence, a heuristic based on GA will be developed in the next section.

3.5. Genetic algorithm-based heuristic

In this section, GA is employed to develop a heuristic, which is allowed to convert the presented problem to the way that near-optimal solutions could be found. The GA-based heuristic shown in Figure 3.3 includes the following main steps: genetic representation; initialization; decoding operator and fitness evaluation; genetic operators consisting of crossover, mutation, and selection; reparation operator; termination criteria.

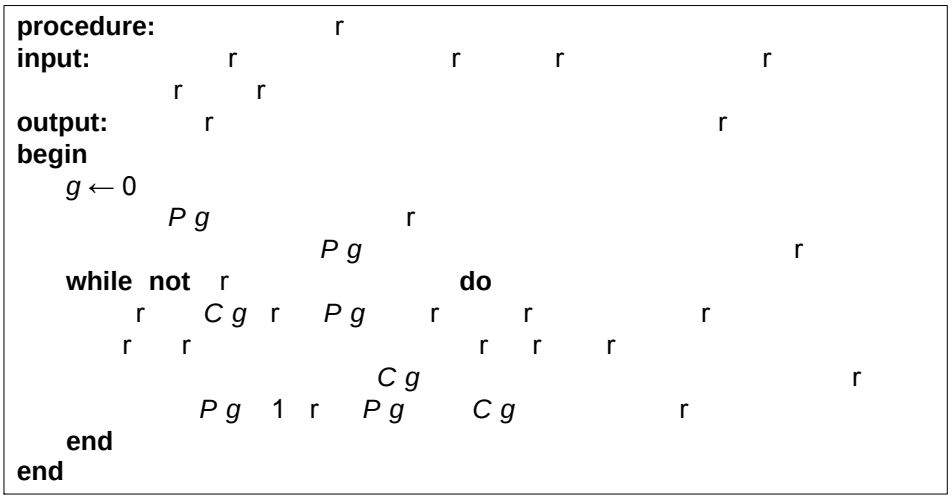

Figure 3.3: Procedure of GA-based heuristic

Genetic representation

For the problem under consideration, a solution can be encoded by a chromosome representing non-preemptive operation sequencing and mobile robot assignment. It means that each gene in the chromosome is made up of two parts. The first part refers to a non-preemptive operation on a specific machine that is assumed to be scheduled at its earliest starting time. The second part identifies a mobile robot carrying out the transportation of that operation. In case the first part of a gene contains the first operation of a task, the second part of that gene will be zero (0) indicating that the first operation of the task does not need to be transported by a mobile robot. It is because the first operation of each task is assumed to be available at a machine at the beginning of the scheduling period as mentioned above. The chromosome length equals the total number of non-preemptive operations of tasks. Figure 3.4 illustrates a chromosome of an example problem with 5 operations of 2 tasks, 3 machines, and 2 mobile robots.

Task	1			2	
Operation	1	2	3	1	2
Machine	M1	M3	M2	M1	M3
Chromosome	21,0	11,0	22,2	12,2	13,1

Figure 3.4: Illustration of a feasible chromosome

Initialization

Random chromosomes are generated for providing solutions to the initial population. A chromosome is constructed of gene by gene. The first part of each gene is assigned an eligible operation (an operation is said to be eligible if all its predecessors are assigned). If that eligible operation is the first operation of any task, zero is assigned to the second part of the gene. Otherwise, one of the mobile robots is randomly chosen to complete the gene. If the chromosome is not yet complete, the eligible set of operations is updated and the process continues. Figure 3.5 shows the procedure of the initialization method.

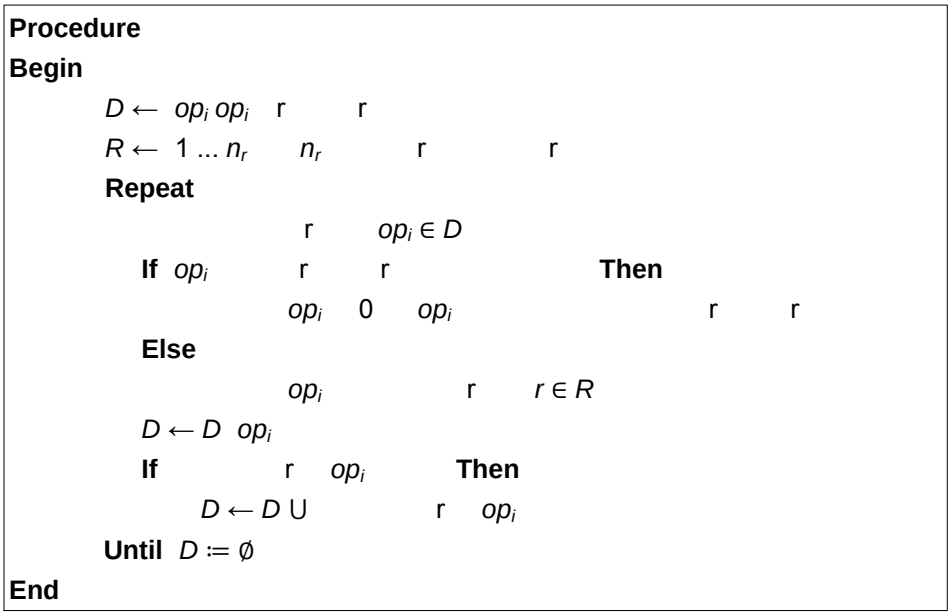

Figure 3.5: Procedure of initialization method

Decoding operator and fitness evaluation

After initialization or reparation routines, chromosomes are decoded and their fitness values are calculated. The decoding operator is mainly composed of the decoding of operation scheduling and decoding of mobile robot assignment. Theoretically, the scheduling of operations is similar to the job shop scheduling problem (Pinedo, 2008). Therefore, the decoding of non-preemptive operation scheduling is carried out under consideration of the predecessor and the last operation processed on the predefined machine of each operation. Furthermore, before the processing of an operation can start, it has to be transported to the predefined machine by an assigned mobile robot. This invokes the decoding of mobile robot assignment. During this step, some information (e.g. total amount of processing time and/or completion time) of the preemptive

operation performed by the mobile robot is also updated. In general, a schedule for preemptive operations is determined through the non-preemptive operation sequencing. Following the decoding operator, the fitness evaluation will take place. The fitness value of a chromosome is equal to the maximum completion time of non-preemptive and preemptive operations. The decoding operator and fitness evaluation are illustrated by the following procedure in Figure 3.6 along with a detailed description below.

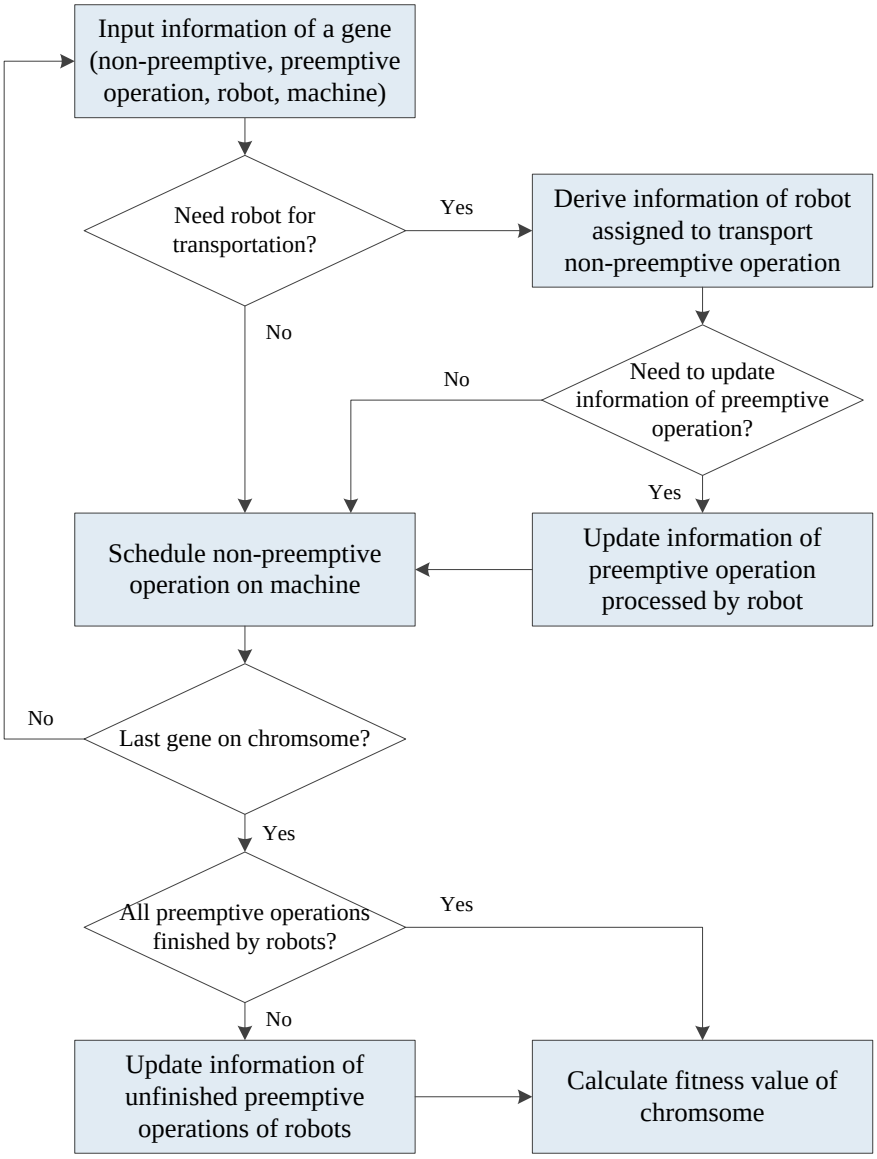

Figure 3.6: Procedure of decoding operator and fitness evaluation

Step 1: Input information of a gene.

op ← operation at gene *i* in a chromosome (non-preemptive operation)

pd ← predecessor of operation *op* in the task sequence

mc ← machine processing operation *op*

mr ← mobile robot transporting operation *op*

om ← operation processed by *mr* (preemptive operation)

Check if there is a need for transportation of operation *op* (check if *mr* ≠ 0). If so, then go to Step 2. If not, then go to Step 4.

Step 2: Derive information of mobile robot *mr* which is assigned to transport operation *op*.

ld ← destination of last trip of mobile robot *mr*

sr ← machine where mobile robot *mr* processes operation *om*

md ← machine processing predecessor *pd*

d_1, d_2, d_3 ← traveling time between *ld* and *sr*, *sr* and *md*, *md* and *mc*

mt: time mobile robot *mr* arrives at machine *md*

ft: time mobile robot *mr* arrives at machine *mc* processing operation *op*

Step 2.1: Check if (finish time of last trip of *mr* + d_1 + d_2) < completion time of *pd*). If so, then *mt* ← completion time of *pd*. If not, then *mt* ← finish time of last trip of *mr* + d_1 + d_2.

Step 2.2*:* Compute *ft* ← *mt* + d_3. Update destination of last trip of *mr* ← *mc*. Update finish time of last trip of *mr* ← *ft*.

Check if *mt* = completion time of *pd*. If so, then go to Step 3. If not, then go to Step 4.

Step 3: Update information of operation *om* processed by mobile robot *mr*

Check if total working time of *mr* < processing time of *om*. If not, then go to Step 4. If so, then check if (processing time of *om* – total working time of *mr*) > completion time of *pd* – (finish time of last trip of *mr* + d_1 + d_2).

If so, then update total working time of *mr* ← total working time of *mr* + completion time of *pd* – (finish time of last trip of *mr* + d_1 + d_2).

Otherwise, then update completion time of *om* ← finish time of last trip of *mr* + d_1 + (processing time of *om* – total working time of *mr*), and total working time of *mr* ← processing time of *om*.

Step 4: Schedule operation *op* on machine *mc*

st: starting time of operation *op*

Step 4.1: Check if the number of scheduled operations on *mc* > 0. If not, then *st* ← 0. If so, then *st* ← completion time of last operation scheduled on *mc*.

Step 4.2: In case $mr \neq 0$, check if $st < ft$ (time mobile robot *mr* arrives at machine *mc*). If so, then $st \leftarrow ft$.

Step 4.3: Compute completion time of operation $op \leftarrow st$ + processing time of *op*. Then check if gene *i* is last gene in the chromosome.

If not, go back to Step 1.

Otherwise, check if all preemptive operations are finished. If not, then go to Step 5. If so, then go to Step 6.

Step 5: Update information of unfinished preemptive operations.

If any mobile robot *mr* has not finished processing operation *om*, then update completion time of *om* ← finish time of last trip of *mr* + d_1 + (processing time of *om* – total working time of *mr*), and total working time of *mr* ← processing time of *om*.

Step 6: Compute the fitness value of the chromosome (maximum completion time of all non-preemptive and preemptive operations).

Genetic operators

Genetic operators mimic the process of heredity of genes to create new offspring at each generation. The operators, in essence, are used to alter the genetic composition of chromosomes and expected to yield improved offspring. Crossover, mutation, and selection are three main genetic operators (Lin et al., 2006).

Crossover operator generates offspring by combining the information contained in the parent chromosomes so that the offspring will have desirable features from their parents. The Roulette-wheel selection is used in the algorithm, which probabilistically selects the parent chromosomes based on their fitness values (Goldberg, 1989). Owing to the nature of the considered minimization problem, the higher the makespan is, the less fitness a chromosome should show. Let $F(p)$ denote the fitness value under the solution represented by parent p, then $F'(p) = \max\{F(p)|1 \leq p \leq N_p\} - F(p)$ where N_p is population size. The expected probability of parent p to be selected is given by $F'_p/\sum F'_p$. Several crossover operators have been used for permutation representation, e.g. two-point crossover, uniform crossover (Abdelmaguid et al., 2004), partially-mapped crossover, order crossover, and position-based crossover (Lin et al., 2006). Although the crossover operators may affect the efficiency of the search process, the quality of solutions is often reasonably close. In the presented experiment, a uniform

crossover operating with probability P_c will be used to generate offspring as described below. Starting from the first operations on the parents, iteratively, one of the parents is randomly selected. The next unconsidered operation of the selected parent becomes the next operation on the first offspring while the next unconsidered operation of the other parent becomes the next operation of the second offspring. If the mobile robot selected for that operation is the same on both parents, then that selection is also made on the child; if not, one of the mobile robots of the parents is randomly chosen. Figure 3.7 depicts the uniform crossover.

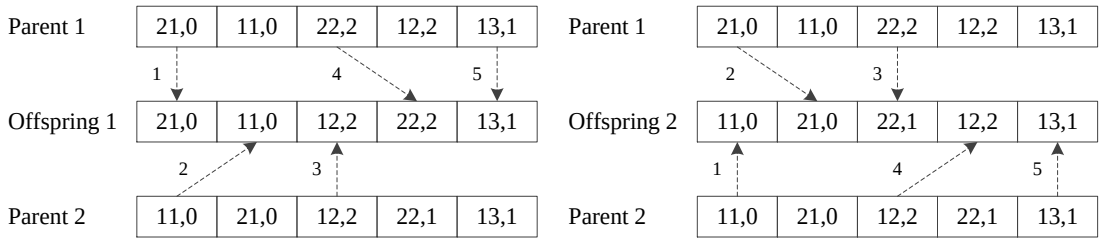

Figure 3.7: Uniform crossover

Mutation is a background operator which produces spontaneous random changes in various chromosomes (Gen and Lin, 2008). For the current encoding method, there are two mutation operators, one for each part of a gene and with a probability P_m. The first mutation operator selects two random positions on a chromosome and swaps the operations with respect to those positions. Note that the chromosome may be infeasible in terms of precedence constraints after this mutation operation. Therefore it has to be adjusted by using the reparation operator presented in the next subsection. The second mutation operator replaces the mobile robot assignment at a gene with one of the mobile robots which is randomly chosen. This may lead to the same mobile robot assignment for a particular gene, and aim to prevent the loss of any good assignment. The mutation operators are shown in Figure 3.8.

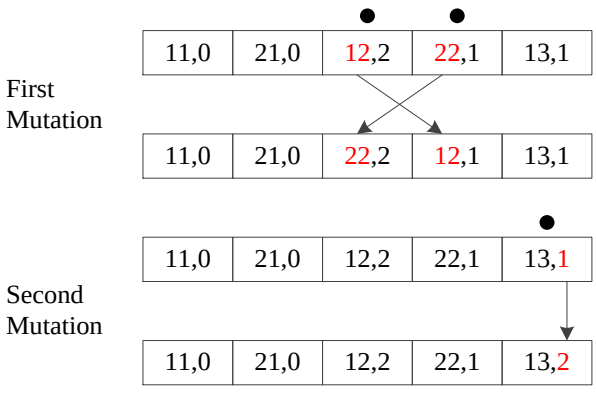

Figure 3.8: Mutation operators

Selection (reproduction) operator is intended to improve the average quality of the population by giving the high quality chromosomes a better chance to get copied into the next generation (Lin et al., 2006). Various selection methods can be applied to this problem. ($\mu + \lambda$) selection is used to choose chromosomes for reproduction. Under this method, μ parents and λ offspring compete for survival and the μ best out of the set of offspring and old parents, i.e. the μ lowest in term of the makespan, are selected as the parents of the next generation. This selection method guarantees that the best solutions up to now are always in the parent generation (Dang et al., 2013b).

Reparation operator

A reparation operator is developed to validate chromosomes with any precedence violations after the mutation operator. This operator involves the exchange of the locations of operations belonging to the same task such that a valid sequence of operations is achieved.

Termination criteria

Termination criteria are used to determine when the GA-based heuristic should be stopped. Note that making decision on which sequence mobile robots and machines should handle tasks is a part of real-time activities of production planners. Therefore, on the one hand, if the best solutions over generations do not converge, the maximum computation time (CT_m) would be used to stop the run. On the other hand, if the best solution does not improve over a number of consecutive generations (G_c), it would not be valuable to continue searching. The up-to-date best solution is then returned as the near-optimal solution. However, it should be noted that high-quality local optima might exist because of the combinatorial nature of the presented problem.

3.6. Numerical experiments and comparison

To examine the performance of the MIP model and GA-based heuristic, a numerical example and computational experiments are conducted in this section. The numerical example has first been created to illustrate the results of both approaches. Various problem instances are then randomly generated and tested in order to provide more persuasive evidence of the performance of the GA-based heuristic. In the experiments, the MIP model has been coded and solved by the mathematical modeling language

ILOG CPLEX, while the GA-based heuristic has been programmed in VB.NET. All the experiments run on a PC having an Intel® Core i5 2.67 GHz processor and 4 GB RAM.

Numerical example

An FMS considered in the numerical example consists of 5 tasks (3 non-preemptive and 2 preemptive tasks) with a total of 9 operations which are carried out on 5 machines. Two mobile robots are employed in transporting the non-preemptive tasks between some machines and processing the preemptive tasks on the other machines. A layout for this example can be seen in Figure 3.1. Table 3.1 gives the assigned machine numbers and processing time. This table also shows the precedence constraints among the operations in each task, e.g. the second operation of task 1 cannot be carried out before the first operation of task 1 is completed. The traveling time of the mobile robots from machines to machines are given in Table 3.2.

Table 3.1: Task description

Task	Operation	Machine	Mobile robot	Processing time (time unit)
1	11	M1	-	28
	12	M3	-	40
2	21	M2	-	32
	22	M1	-	26
	23	M3	-	42
3	31	M2	-	38
	32	M3	-	46
4	41	M4	R1	100
5	51	M5	R2	90

Table 3.2: Traveling time of robots from machines to machines (time unit)

From/To	M1	M2	M3	M4	M5
M1	0	8	8	10	10
M2	10	0	14	8	10
M3	8	12	0	10	12
M4	12	10	12	0	16
M5	10	10	14	10	0

To determine GA parameter settings, pilot runs are carried out to decide on the values of N_p, P_c, P_m, and G_c. Each parameter is tested at three different levels. These are respectively: N_p (50, 100, 200), P_c (0.4, 0.6, 0.8), P_m (0.05, 0.1, 0.2), G_c (50, 100, 200). There are ten observations under each level, and the 30 runs of each parameter are made in random. Moreover, the time for each run is limited to *CTm* of 30. The GA parameters

are chosen according to the best results in terms of the objective value and computation time obtained with these pilot runs. They are as follows: 100, 0.8, 0.1, 100, and 30 for N_p, P_c, P_m, G_c, and CT_m, respectively.

For this numerical example, both MIP model and GA-based heuristic found the optimal solution for the problem. The time required to complete all tasks is 164 (time units), and the schedule is given as: 21,0 - 11,0 - 12,2 - 22,1 - 31,0 - 23,2 - 32,1. The proposed heuristic slightly faster obtains the optimal solution than the MIP model (0.1 second as opposed to 0.7 second). Figure 3.9 shows the solution on Gantt chart. It can be seen from Figure 3.9 that each mobile robot has to interrupt its preemptive task two times to carry out transportation of the non-preemptive tasks. For instance, mobile robot 2 interrupts task 5 the first time to transport task 1 from machine 1 to machine 3, and the second time to transport task 2 also from machine 1 to machine 3. These interruptions consequently divide the duration of each preemptive task (or operation) into three separate parts as shown in the Gantt chart. In general, this numerical example has illustrated the results of the proposed approaches. To be able to illustrate the efficiency of the proposed approaches in more complex cases, larger-sized problems will be investigated in the next section.

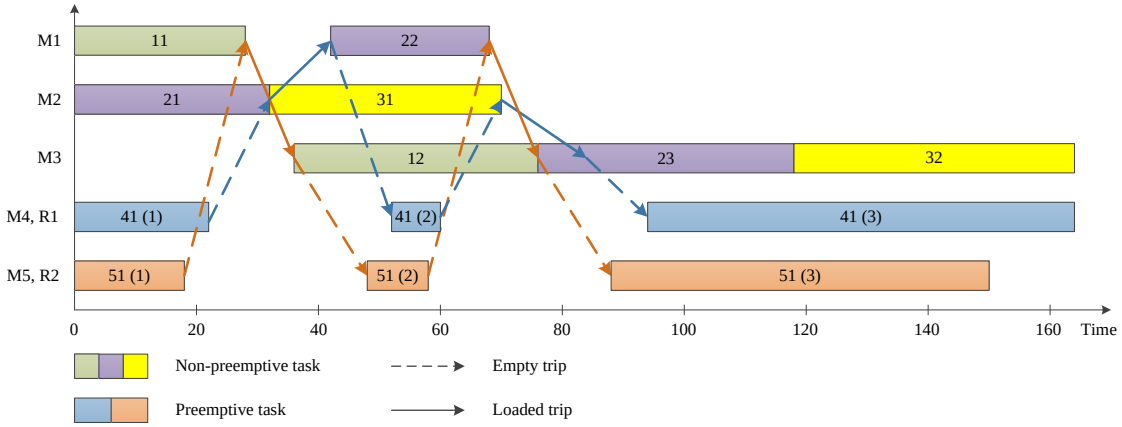

Figure 3.9: Gantt chart for the optimal solution of the numerical example

Computational experiments

In this section, the performance of the GA-based heuristic will be tested on a large number of problem instances. 20 problem instances are generated with different numbers of tasks, operations, machines, mobile robots and other system parameters. The number of all tasks and number of operations in each non-preemptive task are

randomly generated in the ranges of [5, 20] and [2, 6], respectively. The number of machines and mobile robots are respectively distributed within the ranges of [5, 10] and [2, 4]. The processing time of non-preemptive and preemptive operations in time unit are respectively distributed within the ranges of [25, 50] and [100, 200] while the traveling time of mobile robots in time unit are generated in the interval [8, 18]. Note that the time/cost matrices of the generated traveling time should satisfy the triangle inequality. The problem sizes are shown in Table 3.3. The comparisons between the MIP and GA-based heuristic for 20 randomly generated problems are presented in Table 3.4. For each problem instance, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs.

Table 3.3: Problem sizes for 20 randomly generated problems

Problem instance	Problem size				Number of variables			Number of constraints
	Number of tasks	Number of operations	Number of machines	Number of robots	Integer variables	Continuous variables	Total variables	
1	5	12	5	2	74	22	96	154
2	7	25	6	2	412	48	460	839
3	6	16	5	2	144	30	174	296
4	8	30	7	2	592	58	650	1202
5	7	21	5	2	271	40	311	553
6	9	33	6	2	719	64	783	1457
7	10	38	8	2	957	74	1031	1936
8	11	44	7	3	1901	84	1985	3748
9	10	40	6	3	1626	76	1702	3204
10	12	48	7	3	2260	92	2352	4460
11	13	53	8	3	2736	102	2838	5405
12	14	62	7	3	4004	120	4124	7930
13	13	57	9	3	3235	110	3345	6399
14	15	68	8	3	4739	132	4871	9392
15	14	59	9	4	4462	112	4574	8743
16	17	81	10	4	8833	156	8989	17413
17	16	76	9	4	7880	146	8026	15524
18	19	94	10	4	12095	182	12277	23894
19	18	85	9	4	9805	164	9969	19343
20	20	100	10	4	13823	194	14017	27330

Table 3.4: Comparison between MIP and GA-based heuristic for 20 randomly generated problems

Problem instance	MIP		GA-based heuristic			
	Objective value (time unit)	Computation time (second)	Objective value (time unit)		Computation time (second)	
			Average	Standard deviation	Average	Standard deviation
1	248	1.76	248	0	0.25	0.03
2	691	30.00	474	8	0.40	0.08
3	306	5.38	309	2	0.29	0.04
4	818	30.00	534	6	0.46	0.06
5	537	30.00	399	3	0.33	0.05
6	932	30.00	591	10	0.59	0.13
7	1031	30.00	629	12	0.62	0.21
8	1379	30.00	678	18	0.93	0.18
9	1209	30.00	650	16	0.65	0.11
10	1544	30.00	702	18	1.04	0.22
11	1816	30.00	778	20	1.44	0.32
12	2043	30.00	963	24	1.92	0.33
13	1850	30.00	803	19	1.70	0.24
14	2207	30.00	975	24	2.18	0.47
15	1952	30.00	743	22	1.36	0.22
16	2876	30.00	1031	29	2.74	0.54
17	3032	30.00	1015	27	2.04	0.70
18	3836	30.00	1153	29	4.46	0.83
19	3050	30.00	1067	31	3.32	0.54
20	4282	30.00	1239	31	5.33	1.05

It can be observed from Table 3.4 that the GA-based heuristic is superior to the MIP for large problems. The MIP found feasible (not optimal) solutions within the time limit for 18 problem instances. However, these solutions found by the MIP are much worse than those found by the GA-based heuristic. Moreover, the computation time shows that the proposed heuristic was significantly fast in obtaining the best solutions, e.g. approximately 5 seconds for problem instance 20 (the largest-sized problem). For the other problems, both MIP and GA-based heuristic found the optimal solution in problem instance 1 while the objective value found through the proposed heuristic are greater than that found by the MIP in problem instance 3. However, the difference is only about 1% and this is deemed to be an acceptable error. Furthermore, in terms of the objective value, the standard deviation is quite small in comparison with the average. These results provide more persuasive evidence to prove that the GA-based heuristic performs effectively.

3.7. Conclusions

This chapter presents the study results of a problem of simultaneous scheduling of machines and mobile robots in an FMS. Two types of tasks which are preemptive and non-preemptive are taken into account. The mobile robots in this study have capabilities to not only transport non-preemptive tasks similar to material handling devices but also process preemptive tasks by using their manipulation arms. The main novelty of this study lies in the fact that the mobile robot must interrupt their preemptive tasks to carry out transportation of non-preemptive tasks when needed. The objective is to minimize the makespan while satisfying a number of practical constraints. An MIP model is formulated to find optimal solutions for the problem. A GA-based heuristic is then proposed to find near-optimal solutions, which can be applied to deal with large-scale applications. The quality of the proposed heuristic's solutions can be evaluated by using the MIP solutions as reference points. The results from the numerical experiments show that the proposed heuristic is significantly fast to find near-optimal solutions, capable of solving problems of various sizes, and more efficient than the MIP in terms of the objective value when giving the same maximum computation time.

4. Conclusions and future work

Today there is an increasing need for transformable production systems that possess both flexibility and efficiency by using mobile robots. Nevertheless, in order to utilize the mobile robots in an efficient manner, it is important to properly schedule the mobile robots in relation to the production needs. The aim of this thesis is thus to address the scheduling problems of mobile robot(s) at operational levels of manufacturing systems.

Two problems of scheduling mobile robot(s) are taken into account. The first scheduling problem deals with the case of a single mobile robot using its manipulation arm to perform part-feeding tasks on production lines. The simultaneous consideration of hard time windows of part-feeding tasks and capacity constraints resulting in multiple delivery routes in case of a single mobile robot is the main novelty of the first problem. The objective is to minimize the total traveling time of the mobile robot so that it can accomplish all the movements with the smallest consumed amount of battery energy and thereby increase its availability. To solve the first problem, an MIP model is first formulated and then followed by developing a GA-based heuristic with an efficient mechanism to handle variable length of chromosomes/solutions and time windows. The performance of the GA-based heuristic is later evaluated by referring to MIP solutions.

The second scheduling problem expands the scope of the first research by taking into account the simultaneous scheduling of multiple mobile robots and machines with preemptive tasks and non-preemptive tasks in an FMS. The main novelty of the second problem is that the mobile robots can interrupt the preemptive tasks to do transportation of non-preemptive tasks. The objective here is to minimize the makespan and, at the same time, fulfill a number of practical constraints. Similar to the solution methodology in the first problem, a mathematical model is formulated and a GA-based heuristic is developed with an efficient mechanism to handle preemption cases in order to solve the second problem.

The presented results of the numerical experiments in the two scheduling problems show that the use of mathematical models are too time-consuming while the GA-based heuristics are significantly fast to find near-optimal solutions. It can be also concluded that the proposed heuristics have the capability to solve problems of various size, and more efficient than the mathematical models in terms of the objective values when giving the same limited computation time.

The two proposed heuristics support production managers in making decisions at operational levels. For instance, in practice the heuristic method in the case of single mobile robot has been implemented and tested at a real production line of a pump part manufacturing company in Denmark. This heuristic method has been embedded in the Mission Planner and Controller software to schedule the mobile robot tasks in the production line. This heuristic method provided good solutions which helps to maintain operation efficiency on the shop floor level. Furthermore, the two heuristics are efficient to react to unpredicted circumstances such as machine breakdown or robot breakdown. Moreover, they can be applied in a variety of tasks of not only mobile robots but also AGVs.

The research presented in this thesis offers several avenues of further research potentially extending the presented research. One avenue is developing a general model which allows mobile robots to carry out multiple transportations before going back to their processing tasks. Another avenue for further research is to consider mobile robots to transport one or more type(s) of parts to different machines or workstations. Future research can also include a general rescheduling mechanisms based on the obtained schedules and feedback from the mobile robot fleet and shop floor to deal with real-time disturbances such as scraps or lack of materials/parts.

References

Abdelmaguid, T.F., Nassef, O.N., Kamal, B.A. and Hassan, M.F. (2004). A hybrid GA/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research*, 42 (2), 267–281.

Ascheuer, N., Escudero, L.F., Grotschel, M. and Stoer, M.A. (1993). Cutting plane approach to the sequential ordering problem (with application to job scheduling in manufacturing). *SIAM Journal of Optimization*, 3 (1), 25–42.

Askin, R.G. and Stanridge, C.R. (1993). *Modeling and analysis of manufacturing systems*. New York: John Wiley & Sons.

Bilge, Ü. and Ulusoy, G. (1995). A time window approach to simultaneous scheduling of machines and material handling system in a FMS. *Operations Research*, 43 (6), 1058–1070.

Bischoff, R. (2010). Robotics-enabled logistics and assistive services for the transformable factory of the future (TAPAS). EC project.

Blazewicz, J., Eiselt, H.A., Finke, G., Laporte, G. and Weglartz, J. (1991). Scheduling tasks and vehicles in a flexible manufacturing system. *International Journal of Flexible Manufacturing Systems*, 4 (1), 5–16.

Buzacott, J.A. and Yao, D.D. (1986). Flexible manufacturing systems: A review of analytical models. *Management Science*, 32 (7), 890–905.

Carlton, W.B. and Barnes, J.W. (1996). Solving the traveling salesman problem with time windows using tabu search. *IIE Transactions*, 28 (8), 617–629.

Carpaneto, G. and Toth, P. (1980). Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Management Science*, 26 (7), 736–743.

Carpaneto, G., Dell'Amico, M. and Toth, P. (1995). Exact solution of large-scale, asymmetric traveling salesman problems. *ACM Transactions on Mathematical Software*, 21 (4), 394–409.

Caumond, A., Lacomme, P., Moukrim, A. and Tchernev, N. (2009). A MILP for scheduling problems in an FMS with one vehicle. *European Journal of Operational Research*, 199 (3), 706–722.

Chatterjee, S., Carrera, C. and Lynch, L.A. (1996). Genetic algorithms and traveling salesman problems. *European Journal of Operational Research*, 9 (3), 490–510.

Chen, C. and Tseng, C. (1996). The path and location planning of workpieces by genetic algorithms. *Journal of Intelligent Manufacturing*, 7(1), 69–76.

Chen, S.M. and Chien, C.Y. (2011). Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. *Expert Systems with Applications*, 38 (12), 14439–14450.

Choi, I.C., Kim, S.I. and Kim, H.S. (2003). A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem. *Computers and Operations Research*, 30 (5), 773–786.

Dang, Q.V., Nielsen, I. and Steger-Jensen, K. (2012). Mathematical formulation for mobile robot scheduling problem in a manufacturing cell. *In* J. Frick, B. Laugen (Ed.), *APMS 2011, IFIP AICT 384*. Springer-Verlag Berlin Heidelberg, 37–44.

Dang, Q.V. and Nielsen, I. (2013). Simultaneous scheduling of machines and mobile robots. *In* J.M. Corchado et al. (Ed.) *PAAMS 2013 Workshops, CCIS 365*. Springer-Verlag Berlin Heidelberg, 118–128.

Dang, Q.V., Nielsen, I. and Steger-Jensen, K. (2013a). Scheduling a single mobile robot incorporated into production environment. *In* P. Golinska (Ed.), *EcoProduction and Logistics, EcoProduction.* Springer-Verlag Berlin Heidelberg, 185–201.

Dang, Q.V., Nielsen, I., Steger-Jensen, K. and Madsen, O. (2013b). Scheduling a single mobile robot for part-feeding tasks of production lines. *Journal of Intelligent Manufacturing*. DOI: 10.1007/s10845-013-0729-y.

Deroussi, L., Gourgand, M. and Tchernev, N. (2008). A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research*, 46 (8), 2143–2164.

Duša, V. and Barták, R. (2009). Preemptive scheduling with precedences and alternative resources. *Proceedings of Multidisciplinary International Conference on Scheduling: Theory and Applications* (*MISTA*), August 10–12, Dublin, Ireland, 518–530.

Edan, Y., Flash, T., Peiper, U.M., Shmullevich, I. and Sarig, Y. (1991). Near-minimum-time task planning for fruit-picking robots. *IEEE Transactions on Robotics and Automation*, 7 (1), 48–55.

Ganesharajah, T., Hall, N.G. and Sriskandarajah, C. (1998). Design and operational issues in AGV-served manufacturing systems. *Annals of Operations Research*, 76, 109–154.

Gen, M. and Lin, L. (2008). *Network Models and Optimization*. London: Springer-Verlag.

Geng, X., Chen, Z., Yang, W., Shi, D. and Zhao, K. (2011). Solving the travelling salesman problem based on an adaptive simulated annealing algorithm with greedy search. *Applied Soft Computing*, 11 (4), 3680–3689.

Germs, R., Goldengorin, B. and Turkensteen, M. (2012). Lower tolerance-based branch and bound algorithms for the ATSP. *Computer and Operations Research*, 39 (2), 291–298.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization, and machine learning*. New York: Addison-Wesley.

Gornley, M.R. and Eisner, J. (2013). Nonconvex Global Optiminzation for Latent-Variable Models. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, August 4-9, Sofia, Bulgaria, 444–454.

Han, M.H., McGinnis, L.F., Shieh, J.S. and White, J.A. (1987). On sequencing retrievals in an automated storage/retrieval system. *IIE Transactions*, 19 (1), 56–66.

Hasegawa, M., Ikeguchi, T. and Aihara, Kazuyuki (2002). Solving large scale traveling salesman problems by chaotic neurodynamics. *Neural Networks*, 15 (2), 271–283.

Ho, W. and Ji, P. (2004). A hybrid genetic algorithm for component sequencing and feeder arrangement. *Journal of Intelligent Manufacturing*, 15 (3), 307–315.

Hurink, J. and Knust, S. (2002). A tabu search algorithm for scheduling a single robot in a job-shop environment. *Discrete Applied Mathematics*, 119 (1–2), 181–203.

Hvilshøj, M., Bøgh, S., Nielsen, O.S. and Madsen, O. (2012a). Autonomous industrial mobile manipulation (AIMM): past, present and future. *Industrial Robot: An International Journal*, 39(2), 120–135.

Hvilshøj, M., Bøgh, S., Nielsen, O.S. and Madsen, O. (2012b). Multiple part feeding – real-world application for mobile manipulators. *Assembly Automation*, 32 (1), 62–71.

Jerald, J., Asokan, P., Saravanan, R. and Delphin Carolina Rani, A. (2006). Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, 29 (5–6), 584–589.

Khayat, G.E., Langevin, A. and Riopel, D. (2006). Integrated production and material handling scheduling using mathematical programming and constraint programming. *European Journal of Operational Research*, 175 (3), 1818–1832.

Lacomme, P., Larabi, M. and Tchernev, N. (2013). Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Economics*, 143 (1), 24–34.

Lacomme, P., Moukrim, A. and Tchernev, N. (2005). Simultaneous job input sequencing and vehicle dispatching in a single-vehicle automated guided vehicle system: a heuristic branch-and-bound approach coupled with a discrete events simulation model. *International Journal of Production Research*, 43(9), 1911–1942.

Landrieu, A., Mati, Y. and Binder, Z. (2001). A tabu search heuristic for the single vehicle pickup and delivery problem with time windows. *Journal of Intelligent Manufacturing*, 12(5–6), 497–508.

Lin, L., Shinn, S.W., Gen, M. and Hwang, H. (2006). Network model and effective evolutionary approach for AGV dispatching in manufacturing system. *Journal of Intelligent Manufacturing*, 17 (4), 465–477.

Liu, F. and Zeng, G. (2009). Study of genetic algorithm with reinforcement learning to solve the TSP. *Expert Systems with Applications*, 36 (3), 6995–7001.

López-Ibáñez, M. and Blum, C. (2010). Beam-ACO for the travelling salesman problem with time windows. *Computers and Operations Research*, 37 (9), 1570–1583.

Maimon, O., Braha, D. and Seth, V. (2000). A neural network approach for a robot task sequencing problem. *Artificial Intelligence in Engineering*, 14 (2), 175–189.

Miller, D.L. and Pekny, J.F. (1991). Exact solution of large asymmetric traveling salesman problems. *Science*, 251 (4995), 754–761.

Moon, C., Kim, J., Choi, G. and Seo, Y. (2002). An efficient genetic algorithm for the traveling salesman problem with precedence constraints. *European Journal of Operational Research*, 140(3), 606–617.

Moon, C., Seo, Y., Yun, Y. and Gen, M. (2006). Adaptive genetic algorithm for advanced planning in manufacturing supply chain. *Journal of Intelligent Manufacturing*, 17 (4), 509–522.

Ohlmann, J.W. and Thomas, B.W. (2007). A compressed-annealing heuristic for the traveling salesman problem with time windows. *INFORMS Journal on Computing*, 19 (1), 80–90.

Pinedo, M.L. (2008). Scheduling: Theory, Algorithms, and Systems. New York: Springer.

Potvin, J.Y. (1996). Genetic algorithms for the traveling salesman problem. *Annals of Operation Research*, 63 (3), 339–370.

Ratliff, H.D. and Rosenthal, A.S. (1983). Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem. *Operations Research*, 31 (3), 507–521.

Reddy, B.S.P. and Rao, C.S.P. (2006). A hybrid multi-objective GA for simultaneous scheduling of machines and AGVs in FMS. *The International Journal of Advanced Manufacturing Technology*, 31 (5–6), 602–613.

Reinet, G. (1991). TSPLIB – A traveling salesman problem library. *ORSA Journal on Computing*, 3 (4), 376–384.

Reuther, H., Simon, B. and Mads, H. (2010). 3D Simulation used for evaluating the use and implementation of the mobile manipulator "Little Helper" at Grundfos A/S. *In* H. Hvolby, *Proceedings of the 12th International Modern Information Technology in the Innovation Processes of the Industrial Enterprises* (*MITIP*) *Conference*, August 29-31, Aalborg: Centre for Logistics, Aalborg University, 9–18.

Schuler, J. (1987). *Integration von Förder- und Handhabungseinrichtungen*. Berlin: Springer.

Silver, E.A., Pyke, D.F. and Peterson, R. (1998). *Inventory management and production planning and scheduling*. New York: John Wiley & Sons.

Snyder, L.V. and Daskin, M.S. (2006). A random-key genetic algorithm for the generalized traveling salesman problem. *European Journal of Operational Research*, 174 (1), 38–53.

Soylu, M., Özdemirel, N.E. and Kayaligil, S. (2000). A self-organizing neural network approach for the single AGV routing problem. *European Journal of Operational Research*, 121 (1), 124–137.

Suárez, R. and Rosell, J. (2005). Feeding sequence selection in a manufacturing cell with four parallel machines. *Robotics and Computer-Integrated Manufacturing*, 21 (3), 185–195.

Syslo, M.M., Deo, N. and Kowalik, J.S. (1983). *Discrete optimization algorithms: with Pascal programs*. New Jersey: Prentice-Hall.

Toth, P. and Vigo, D. (2002). *The vehicle routing problem*. Philadelphia: SIAM.

Tsai, C.F., Tsai, C.W. and Tseng, C.C. (2004). A new hybrid heuristic approach for solving large traveling salesman problem. *Information Sciences*, 166 (1–4), 67–81.

Tsujimura, Y. and Gen, M. (1999). Parts loading scheduling in a flexible forging machine using an advanced genetic algorithm. *Journal of Intelligent Manufacturing*, 10 (2), 149–159.

Turkensteen, M., Ghosh, D., Goldengorin, B. and Sierksma, G. (2008). Tolerance-based branch and bound algorithms for the ATSP. *European Journal of Operational Research*, 189 (3), 775–788.

Ulusoy, G. and Bilge, Ü. (1993). Simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research*, 31 (12), 2857–2873.

Ulusoy, G., Sivrikaya-Serifoglu, F. and Bilge, Ü. (1997). A genetic algorithm approach to the simultaneous scheduling of stations and automated guided vehicles. *Computers & Operations Research*, 24 (4), 335–351.

Xing, L.N., Chen, Y.W., Yang, K.W., Hou, F., Shen, X.S. and Cai, H.P. (2008). A hybrid approach combining an improved genetic algorithm and optimization strategies for the asymmetric traveling salesman problem. *Engineering Applications of Artificial Intelligence*, 21(8), 1370–1380.

Yun, Y.S. (2002). Genetic algorithm with fuzzy logic controller for preemptive and non-preemptive job-shop scheduling problems. *Computers and Industrial Engineering*, 43 (3), 623–644.

Zacharia, P.Th. and Aspragathos, N.A. (2005). Optimal robot task scheduling based on genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, 21(1), 67–79.

Paper B

Mathematical formulation for mobile robot scheduling problem in a manufacturing cell

This paper has been published in APMS 2011, IFIP AICT 384.
Springer-Verlag Berlin Heidelberg, pp. 37–44

Mathematical Formulation for Mobile Robot Scheduling Problem in a Manufacturing Cell

Quang-Vinh Dang, Izabela Nielsen, and Kenn Steger-Jensen

Department of Mechanical and Manufacturing Engineering, Aalborg University,
Fibigerstræde 16, 9220 Aalborg, Denmark
{vinhise,izabela,kenn}@m-tech.aau.dk

Abstract. This paper deals with the problem of finding optimal feeding sequence in a manufacturing cell with feeders fed by a mobile robot with manipulation arm. The performance criterion is to minimize total traveling time of the robot in a given planning horizon. Besides, the robot has to be scheduled in order to keep production lines within the cell working without any shortage of parts fed from feeders. A mixed-integer programming (MIP) model is developed to find the optimal solution for the problem. In the MIP formulation, a method based on the (s, Q) inventory system is applied to define time windows for multiple-part feeding tasks. A case study is implemented at an impeller production line in a factory to demonstrate the result of the proposed MIP model.

Keywords: Scheduling, Mobile Robot, MIP, Feeding Sequence.

1 Introduction

Production systems nowadays range from fully automated to strictly manual. While the former is very efficient in high volumes but less flexible, the latter is very flexible but less cost-efficient. Therefore, manufactures visualize the need for transformable production systems that combines the best of both worlds by using new assistive automation and mobile robots. With embedded batteries and manipulation arms, mobile robots are more flexible to perform certain tasks such as transporting and feeding materials, machine tending, pre-assembly or quality inspection at different workstations of production lines. These tasks have such relatively low level of complexity that mobile robots are able to take over. Besides, using mobile robots can lead to less energy usage or less tool-changing costs than commonly industrial robots attached to a fixed surface. These advantages pave the way for mobile robot to be implemented in the transformable production systems. Within the scope of this study, a given problem is particularly considered for a single mobile robot which will automate multiple-part feeding tasks by not only transporting but also collecting containers of parts and emptying them into the feeders needed. However, to utilize mobile robots in an efficient manner requires the ability to properly schedule these feeding tasks. Hence, it is important to plan in which sequence mobile robots process feeding operations so that they could effectively work while satisfying a number of technological constraints.

J. Frick and B. Laugen (Eds.): APMS 2011, IFIP AICT 384, pp. 37–44, 2012.

Robot scheduling problem which is NP-hard has attracted interest of researchers in recent decades. Dror and Stulman [4] dealt with the problem of optimizing one-dimensional robot's service movements. Crama and van de Klundert [1] considered the flow shop problem with one transporting robot and one type of product to find shortest cyclic schedule for the robot. Afterwards, they demonstrated that the sequence of activities whose execution produces one part yields optimal production rates for three-machine robotic flow shops [2]. Crame et al. [3] also presented a survey of cyclic robotic scheduling problem along with their existing solution approaches. Kats and Levner [5], [6] considered m-machine production line processing identical parts served by a mobile robot to find the minimum cycle time for 2-cyclic schedules. Maimon et al. [7] introduced a neural network method for a material-handling robot task-sequencing problem. Suárez and Rosell [8] dealt with the particular real case of feeding sequence selection in a manufacturing cell consisting of four parallel identical machines. Several feeding strategies and simulation model were built to select the best sequence. Most of the work and theory foundation considered scheduling robots which are usually inflexible, move on prescribed path and repeatedly perform a limited sequence of activities. There is still lack of scheduling a free-ranging mobile robot which is able to move around within a manufacturing cell to process multiple-part feeding tasks consisting of collecting, transporting, and delivering containers of parts to feeders. The scheduling problem becomes interesting as the robot has been coordinated to manufacturing so that robot's services maintain production in the lines. Therefore, in this paper we focus on scheduling a single mobile robot for multiple-part feeding tasks whose time windows could be determined based on the inventory system (s, Q) as well as predefined maximum and minimum levels of parts in feeders.

The remainder of this paper is organized as follows: in the next section, problem statement is described while the mathematical model is formulated in Section 3. A case study is investigated to demonstrate the result of the proposed model in Section 4. Finally, conclusions are drawn in Section 5.

2 Problem Description

Fig. 1 below shows a typical layout of the manufacturing cell. In particular, the work is developed for a real cell that produces parts for the pump manufacturing industry at a factory in Denmark. The cell consists of a central storage known as a part supermarket, a single mobile robot, and several production lines including multiple machines which are fed by multiple feeders. An operator is responsible to put parts into small load carriers (SLCs) which are placed in the storage. The robot will retrieve and carry several SLCs containing parts from the storage, move to feeder locations, feed all parts inside each SLC to each feeder, then return to the storage to unload all empty SLCs and take filled SLCs. Because of the limitation on capacity, the feeders have to be served a number of times in order maintain production without any shortage of parts. The mobile robot thus has a set of feeding tasks to carry out during a given planning horizon.

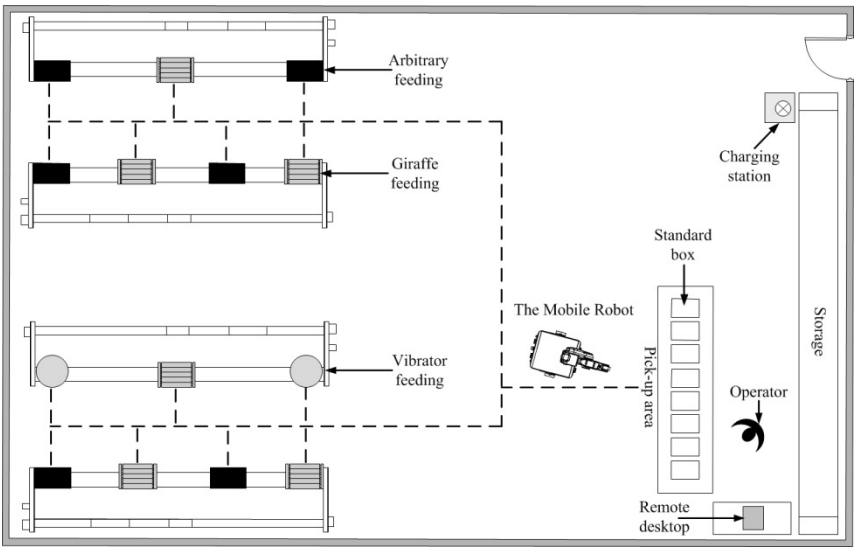

Fig. 1. Layout of the manufacturing cell

To enable the construction of a multiple-part feeding schedule of the mobile robot, the following assumptions are made: an autonomous mobile robot is considered in disturbance free environment; the robot is able to carry one or several SCLs at a time; all tasks are periodic, independent, and assigned to the same robot; working time and traveling time of the robot between any locations, and consuming rates of parts in feeders are known; all feeders of machines have to be fed up to maximum levels and the robots starts from the storage at the initial stage. In order to accomplish all the movements with a smallest consumed amount of battery energy, the total traveling time of the robot is an important objective to be considered. Hence, it is important to determine in which way the robot should feed the feeders of machines in order to minimize its total traveling time within the manufacturing cell while preventing the production lines from stopping working.

3 Mathematical Formulation

In this study, a mix-integer programming (MIP) model is developed to determine an optimal route of the mobile robot visiting a number of locations to process multiple-part feeding tasks. The model is inspired by well-known traveling salesman problem [9] and the (s, Q) inventory system [10]. The latter is applied to define time windows for the feeding tasks. In practice, the MIP model can be applied to small-scale problems with a few numbers of feeders and short planning horizon. Under these scenarios, the MIP model is reasonably fast to give exact optimal solutions, which can be used as reference points to quantify the scale of benefits achieved by a meta-heuristic method further developed. Notations, time windows, and a formulation for the MIP model are extensively described in the following subsections.

3.1 Notations

N : set of all tasks ($N = \{0, 1, 2, \ldots, n\}$ where 0: task at the storage)
n_i : number of times task i has to be executed
R : set of all possible routes ($R = \{1, 2, \ldots, R_{max}\}$ where $R_{max} = \Sigma\, n_i, \forall i \in N \setminus \{0\}$)
e_{ik} : k-th release time of task i
d_{ik} : k-th due time of task i
p_i : periodic time of task i
w_i : working time of robot at task i location
t_{ij} : traveling time of robot from task i location to task j location
c_i : consuming rate of parts in feeder at task i location
v_i : minimum level of parts in feeder at task i location
u_i : maximum level of parts in feeder at task i location
Q : maximum number of SLCs could be carried by robot
T : planning horizon

Decision variables:

$$x_{ik}^{jlr} = \begin{cases} 1 & \text{if robot travels from } k\text{-th task } i \text{ location to } l\text{-th task } j \text{ location in the route } r \\ 0 & \text{otherwise} \end{cases}$$

y_{ik} : route number to which k-th task i belongs
s_{ik} : k-th starting time of task i

3.2 Time Windows

Time windows of multiple-part feeding tasks of the mobile robot could be determined as shown in Equation (1), (2), and (3) below.

$$p_i = (u_i - v_i)c_i, \forall i \in N \setminus \{0\} \tag{1}$$

$$e_{ik+1} = e_{ik} + p_i, \forall i \in N \setminus \{0\}, k = 1 \div n_i \tag{2}$$

$$d_{ik} = e_{ik} + (v_i - 0)c_i, \forall i \in N \setminus \{0\}, k = 1 \div n_i \tag{3}$$

Task for feeder i whose periodic time is calculated as Equation (1) has a number of times/executions $n_i = \lfloor T / p_i \rfloor$ to be performed. The release time of an execution of task i is set when the number of parts inside feeder i falls to a certain level v_i (Equation (2)); while the due time of an execution of task i is defined when there are no parts in feeder i (Equation (3)).

3.3 Mixed-Integer Programming Model

Objective function:

$$\min \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N} \sum_{l=1}^{n_j} \sum_{r \in R} t_{ij} x_{ik}^{jlr} \tag{4}$$

$$e_{ik} \le s_{ik} \le d_{ik} \qquad \forall i \in N \setminus \{0\}, k = 1 \div n_i \tag{5}$$

$$\sum_{j\in N\setminus\{0\}}\sum_{l=1}^{n_j} x_{01}^{jl1} = 1 \tag{6}$$

$$\sum_{j\in N\setminus\{0\}}\sum_{l=1}^{n_j}\sum_{r\in R} x_{01}^{jlr} \le 1 \tag{7}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i} x_{ik}^{ikr} = 0 \qquad \forall r \in R \tag{8}$$

$$\sum_{r\in R} x_{ik}^{jlr} \le |Z| \text{-} 1 \qquad \forall i, j \in N, k = 1 \div n_i, l = 1 \div n_j, i \ne j, Z \subseteq Z_T, Z \ne \Phi \tag{9}$$

$$\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} x_{ik}^{jlr} = 1 \qquad \forall i \in N\setminus\{0\}, k = 1 \div n_i \tag{10}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} x_{ik}^{jlr} = 1 \qquad \forall j \in N\setminus\{0\}, l = 1 \div n_j \tag{11}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N\setminus\{0\}}\sum_{l=1}^{n_j} x_{ik}^{jlr} \le Q \qquad \forall r \in R \tag{12}$$

$$s_{ik} + \left(w_i + t_{ij}\sum_{r\in R} x_{ik}^{jlr}\right) - L\left(1 - \sum_{r\in R} x_{ik}^{jlr}\right) + \left(y_{jl} - y_{ik}\right) \times \left(t_{i0} + w_0 + t_{0j} - t_{ij}\right) \le s_{jl} \tag{13}$$
$$\forall i, j \in N, k = 1 \div n_i, l = 1 \div n_j$$

$$y_{jl} = \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} r \times x_{ik}^{jlr} \qquad \forall j \in N\setminus\{0\}, l = 1 \div n_j \tag{14}$$

$$y_{jl} \ge y_{ik}\sum_{r\in R} x_{ik}^{jlr} \qquad \forall i, j \in N, k = 1 \div n_i, l = 1 \div n_j \tag{15}$$

$$x_{ik}^{jlr} \in \{0,1\} \qquad \forall r \in R, \forall i, j \in N, k = 1 \div n_i, l = 1 \div n_j \tag{16}$$

y_{ik}: positive integer variable $\qquad \forall i \in N, k = 1 \div n$ (17)

The objective function (4) minimizes the total traveling time of the robot. Constraint (5) ensures that starting time of an execution of a task satisfies its time window. Constraints (6) and (7) indicate that the robot starts from the storage at the initial stage. Constraint (8) prevents the robot repeating an execution of a task. Constraint (9) eliminates the sub-tours among executions of tasks, where Z is a subset of Z_T, where Z_T is a set of all executions of tasks at feeders and the storage, and Φ denotes and empty set. Constraints (10) and (11) force an execution of a task in one route to be done exactly one. Constraint (12) forbids the robot to feed more SLCs than the maximum number of SLCs Q it allows to carry. Constraint (13) handles the traveling time requirements between any pair of executions of tasks, where L is a given sufficiently large constant. In case two executions of the same task or different tasks are connected but they are not in the same route, the robot should visit the storage to unload empty SLCs and load filled ones. Constraint (14) assigns an execution of a task to a route and constraint (15) guarantees the ascending sequence of route numbers for executions of tasks. Constraints (16) and (17) imply the types of variables.

4 Case Study

To examine performance of the MIP model, a case study is investigated at the CR factory at Grundfos A/S. The chosen area for this case study is the CR 1-2-3 impeller production line which produces impellers for industrial pumps. The CR line consists of four feeders that have to be served by the mobile robot. These feeders are indexed from 1 to 4 and named Back Plate, Van Feeder 1, Van Feeder 2, and Front Plate respectively. Besides, different feeders are filled by different kinds of parts, namely back plates for feeder 1, vanes for feeder 2 and 3, front plates for feeder 4. On the CR line, a number of vanes are welded together with back and front plates to produce an impeller. Fig. 2 below particularly illustrates the aforementioned production area where the proposed model has been implemented in the factory.

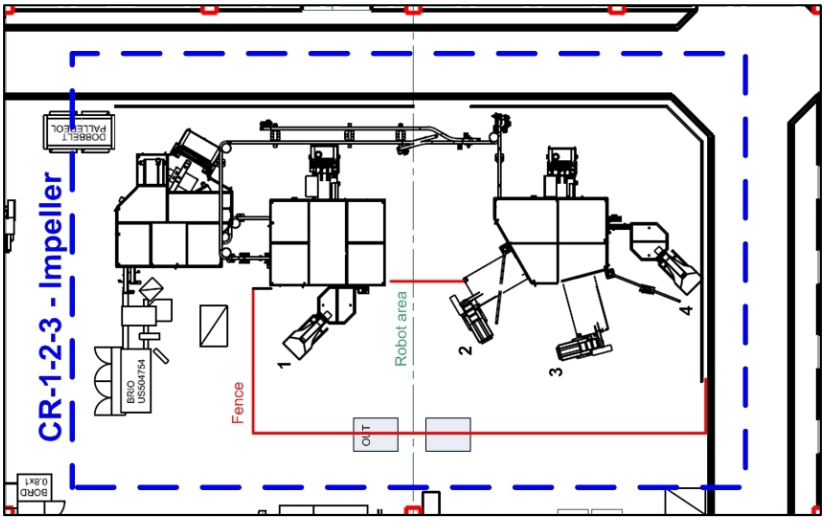

Fig. 2. CR 1-2-3 impeller production line

The maximum number of SLCs carried by the robot is 3. The average number of parts per SLC fed to feeder 1 or 4 is 125 (approximately 2 kg/SLC), while the average number of parts per SLC fed to feeder 2 or 3 is 1100 (approximately 1 kg/SLC). The maximum levels, minimum levels, consuming rates of parts, and working time of the robot are given in Table 1, while Table 2 shows traveling time of the robot from one location of a task to another (feeder 0 means the central storage).

Table 1. Maximum, minimum levels, consuming rates, and working time of robot at feeders

Feeder	0	1	2	3	4
Maximum level (part)	-	250	2000	2000	250
Minimum level (part)	-	125	900	900	125
Consuming rate (s/part)	-	4.5	1.5	1.5	4.5
Working time of robot (s)	90	42	42	42	42

Table 2. Traveling time of robot from one location to another

Traveling time (s)	0	1	2	3	4
0	0	43	33	27	48
1	46	0	29	30	53
2	32	34	0	36	42
3	29	26	35	0	40
4	47	60	46	39	0

The case study has been investigated during approximately 50 minutes because of the limitation on robot batteries. The MIP model has been coded in the mathematical modeling language ILOG OPL 3.6. The problem of case study has been run on a PC having an Intel® Core i5 2.3 GHz processor and 4 GB RAM. The optimal solution obtained is given as: 0 – 1 – 4 – 4 – 0 – 1 – 1 – 4 – 0 – 1 – 2 – 4 – 0 – 3 – 0 , with total traveling time being 503 seconds which makes up 16.4 % of the total time. With 4040 decision variables, the computational time for this case using the proposed model is 4305 seconds. The detailed solution is shown in Table 3 and Fig. 3 below.

Table 3. Detailed optimal solution of the case study

Task	Feeder	Index of execution	Starting time	Route
1	1	1	1030.0	1
2	4	1	1125.0	1
3	4	2	1375.5	1
4	1	2	1687.5	2
5	1	3	2155.0	2
6	4	3	2250.0	2
7	1	4	2549.0	3
8	2	1	2620.0	3
9	4	4	2704.0	3
10	3	1	3000.0	4

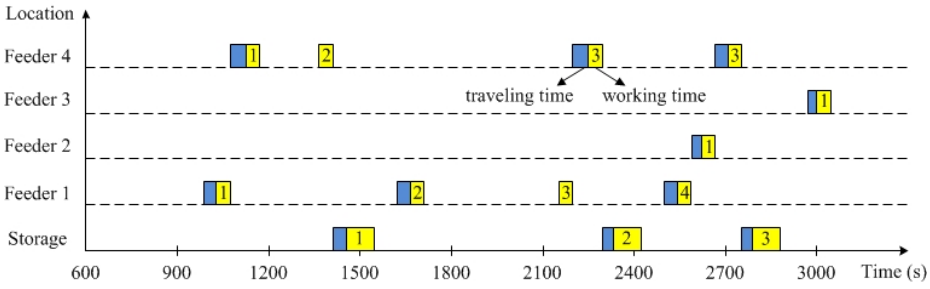

Fig. 3. Gantt chart for the optimal solution of the case study

The above optimal solution is an initial schedule for the robot. That schedule serves as an input to a program called Mission Planner and Control (MPC) which is implemented in VB.NET. The MPC program is accessed using XML-based TCP/IP

communication to command and get feedbacks from the robot. During the practical feeding operations at CR 1-2-3 impeller production line, the initial schedule was executed in sequence and it prevented all of feeders running out of parts. Hence, the CR line can keep producing impellers without shortage of parts fed from feeders.

5 Conclusions

In this paper, a new problem of scheduling a single mobile robot for multiple-part feeding tasks in a manufacturing cell is studied. To accomplish all tasks within allowable limit of battery capacity, it is important for planners to determine optimal feeding sequence to minimize total traveling time of the mobile robot while considering specific features of the robot and a number of technological constraints. An MIP model is developed to find optimal solution for the problem. A particular real case of the impeller production line composing of four feeders is described to show result of the proposed model. The result was quite properly applied during practical feeding operations and it demonstrated that all feeders had no shortage of parts. For further research, the complexity of the problem will increase when considering a larger number of feeders and/or longer planning horizon. Hence, a meta-heuristic method will be taken into account for solving large-scale mobile robot scheduling problems. Besides, re-scheduling mechanisms based on obtained schedules and feedback from the shop floor will be developed to deal with real-time disturbances.

Acknowledgments. This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

References

1. Crama, Y., van de Klundert, J.: Cyclic Scheduling of Identical Parts in a Robotic Cell. Oper. Res. 45, 952–965 (1997)
2. Crama, Y., van de Klundert, J.: Cyclic Scheduling in 3-machine Robotic Flow Shops. J. Sched. 2, 35–54 (1999)
3. Crama, Y., Kats, V., van de Klundert, J., Levner, E.: Cyclic Scheduling in Robotic Flowshops. Ann. Oper. Res. 96, 97–124 (2000)
4. Dror, M., Stulman, A.: Optimizing Robot's Service Movement: a One Dimensional Case. Comput. Ind. Eng. 12, 39–46 (1987)
5. Kats, V., Levner, E.: Parametric Algorithms for 2-cyclic Robot Scheduling with Interval Processing Times. J. Sched. 14, 267–279 (2011)
6. Kats, V., Levner, E.: A Faster Algorithm for 2-cyclic Robotic Scheduling with a Fixed Robot Route and Interval Processing Times. Eur. J. Oper. Res. 209, 51–56 (2011)
7. Maimon, O., Braha, D., Seth, V.: A Neural Network Approach for a Robot Task Sequencing Problem. Artif. Intell. Eng. 14, 175–189 (2000)
8. Suárez, R., Rosell, J.: Feeding Sequence Selection in a Manufacturing Cell with Four Parallel Machines. Robot Comput. Integrated Manuf. 21, 185–195 (2005)
9. Toth, P., Vigo, D.: The Vehicle Routing Problem. SIAM, Philadelphia (2002)
10. Silver, E.A., Pyke, D.F., Peterson, R.: Inventory Management and Production Planning and Scheduling. John Wiley & Sons, New York (1998)

Paper C

Scheduling a single mobile robot incorporated into production environment

This paper has been published in EcoProduction & Logistics, EcoProduction. Springer-Verlag Berlin Heidelberg, pp. 185–201

Scheduling a Single Mobile Robot Incorporated into Production Environment

Quang-Vinh Dang, Izabela Ewa Nielsen and Kenn Steger-Jensen

Abstract Eco-production and logistics with environmental consciousness are playing a larger role in manufacturing firms. They involve scheduling, planning, developing and implementing manufacturing processes and technologies that are required not only to keep productivity high but also to respond to the challenges of issues such as energy conservation and pollution preventions. Facing the central tension between manufacturing and environmental drivers is difficult, but critical to develop new technologies, particularly mobile robots, that can be incorporated into production to achieve holistic solutions. This chapter deals with the problem of finding optimal operating sequence in a manufacturing cell of a mobile robot with manipulation arm that feeds materials to feeders. The "Bartender Concept" is discussed to show the cooperation between the mobile robot and industrial environment. The performance criterion is to minimize total traveling time of the robot with the smallest consumed amount of battery energy in a given planning horizon. A mixed-integer programming (MIP) model is developed to find the optimal solutions for the problem. Two case studies are implemented at an impeller production line to demonstrate the results of the proposed MIP model.

Q.-V. Dang · K. Steger-Jensen
Department of Mechanical and Manufacturing Engineering,
Aalborg University, Fredrik Bajers Vej 5, Aalborg, 9220 Denmark
e-mail: vinhise@m-tech.aau.dk

K. Steger-Jensen
e-mail: kenn@m-tech.aau.dk

I. E. Nielsen (✉)
Department of Mechanical and Manufacturing Engineering, Aalborg University,
Fibigerstraede 16, Aalborg East, 9220 Denmark
e-mail: izabela@m-tech.aau.dk

P. Golinska (ed.), *EcoProduction and Logistics*, EcoProduction,
DOI: 10.1007/978-3-642-23553-5_12,

1 Introduction

Environmental issue is rapidly emerging as one of the most important topics for manufacturing decisions (Azzone and Noci 1998). This is mainly driven by the escalating deterioration of the environment, for examples diminishing raw material resources, overflowing waste sites and increasing levels of pollution (Gungor and Gupta 1999). In the industry nowadays, managers take account of improvements in environmental performance one of the basic competitive edges, alongside lower costs and higher quality. Environmentally conscious manufacturing, or Eco-production, is the transformation of materials into useful products through a value-added process that simultaneously enhances economic well-being and sustains environmental quality (Darnall et al. 1994). It involves planning, developing, and implementing manufacturing processes and technologies that are required to not only keep productivity high but also respond to challenges a variety of issues, such as energy conservation, pollution preventions, and avoidance of ecological degradation (Sarkis 1995). Besides, today's production systems range from fully automated to strictly manual. While the former is very efficient in high volumes but less flexible, the latter is reversed. Hence, manufactures visualize the need for new production systems that combines the best of both worlds and considers environmental impacts by using eco-friendly production technologies such as new assistive automation and autonomous mobile robots.

Mobile robots have the capability to move around within their environment, and they are not fixed to one physical location. With embedded batteries and manipulation arms, mobile robots are more flexible to perform certain tasks such as transporting and feeding materials, machine tending, pre-assembly or quality inspection at different workstations in a manufacturing cell. Moreover, using mobile robots can lead to production efficiency gains (for example less energy usage than commonly industrial robots attached to a fixed surface), better housekeeping, safer and cleaner facilities. These superior advantages of these robots pave the way for meeting the need of the aforementioned new production systems. In this chapter, a given problem is particularly considered for mobile robots with manipulation arms, which will automate extended logistic tasks by not only transporting but also collecting containers of parts and emptying them into the place needed. Feeding operation studied here is a kind of extended logistic tasks. However, to operate mobile robots incorporated into production environment requires better inventory control with a central warehouse where supplies materials or parts for feeding tasks. Furthermore, to utilize mobile robots in an energy-efficient manner requires the ability to properly schedule this kind of tasks. Therefore, it is important to implement a specific concept for better working and environmental order as well as to plan in which sequence the robots process feeding operations so that they could effectively work while satisfying a number of technological constraints.

In the survey of the literature on production considering environmental impacts, Sarkis (1995) presented the environmental problems pertaining to manufacturing

and operation management. The general framework on how to manage environmentally conscious issues in a manufacturing company is develop and discussed. Sarkis and Rasheed (1995) also identified and described the basic elements of an environmental conscious manufacturing strategy and techniques to enable businesses to pursue that strategy. Azzone and Noci (1998) introduced the contingency framework, which analyzed whether and how different "green" manufacturing must be deployed and accessed as well. Gungor and Gupta (2010) discussed the evolution of environmentally conscious manufacturing, which has taken place in the last decade, and new areas, which have come into focus during this time. Srinivasan and Sheng (1999a, b) presented a formalized approach towards integrating environmental factors in process planning which includes micro-planning and macro-planning. In micro-planning, process, parameter and common setups were selected for the individual features, while in macro-planning interactions between features are examined. Krishnan and Sheng (2000) developed an automatic process-planning agent for CNC machining for minimal environmental impact along major categories. The process planner is designed to interact with conventional generative planner as an advisory environmental agent. Jin and Balasubramaniam (2003) integrated the environmentally benign process planning with fuzzy set theories as fuzzy environmental processing planning can handle the vagueness and impreciseness associated with the data and can eliminate subjective decisions. Nielsen et al. (2010) suggested that a widely distributed and semi-structured network of waste producing and waste collecting/processing enterprises could improve their planning both by a proposed Decision Support System based on Constrained Logic Programming, and by implementing RFID technology to update and validate information. Besides, in the literature on production considering robot scheduling problem, which is NP-hard, this topic has attracted interest of researchers in recent decades. Dror and Stulman (1987) dealt with the problem of defining the best movement decision for a one-dimensional service robot. The simple simulation experiment was presented to illustrate an increase in production output with the help of sophisticated decision rules for robot's movement. Chen and Su (1995) addressed the problem of scheduling single-gripper gantry robots whose optimal schedule was developed by analyzing the cycle time formula for two and three-workstation production lines and then by extending the result for the problem of scheduling a production line with multiple workstations. Crama and van de Klundert (1997) considered the flow shop problem with one transporting robot and one type of product to find shortest cyclic schedule for the robot. The dynamic programming approach was developed to solve the problem in polynomial time. Afterwards, they demonstrated that the sequence of activities whose execution produces one part yields optimal production rates for three-machine robotic flow shops (Crama and van de Klundert 1999). Crama et al. (2000) also presented a survey of cyclic robotic scheduling problems, model for such problems, complexity of solving these problems, and the existing solution approaches. Kats and Levner (2010, 2011) considered an m-machine production line processing identical parts served by a mobile robot to find the minimum cycle time for 2-cyclic schedules. The strongly polynomial algorithm of time complexity

and the improved algorithm with reduced complexity were proposed. Maimon et al. (2000) introduced a neural network method for the problem of sequencing tasks comprising loading and unloading of parts into and from the machines by material-handling robot. Suárez and Rosell (2005) analyzed the particular real case of feeding sequence selection in a manufacturing cell consisting of four parallel identical machines working alternately on two pallets, fed by one robot. Several feeding strategies and simulation model were built to select the best sequence. Most of the work and theory foundation considered scheduling robots, which are usually inflexible, move on prescribed path and repeatedly perform a limited sequence of activities. Another figure of a mobile robot that has been extensively studied and in common use today is automated/automatic guided vehicle, but it still follows markers, reflectors or guided wires on the floor. There is still lack of scheduling free-ranging mobile robots, which are able to move around within a manufacturing cell to process extended logistic tasks consisting of collecting, transporting and delivering parts or components right to places. The problem becomes interesting, as the robot has to be incorporated into manufacturing so that robot services maintain production in the lines. The complexity of the problem rapidly rises to be NP-hard when the robot has to serve more workstations (in one or several production lines) and/or work in a longer planning horizon. In addition, environmentally conscious issues, for examples energy conservation and warehouse management, have not intensively discussed in the production coordinated with mobile robots. Therefore, in this chapter we consider implementing the "Bartender Concept" for better working environment of the mobile robot and scheduling that kind of robot for feeding tasks, which could be pre-determined based on maximum and minimum levels of parts in feeders, so that all of tasks are completed with a smallest consumed amount of battery energy.

The remainder of this chapter is organized as follows: in the next section, the "Bartender Concept" is discussed to show the cooperation between the robot and industrial environment. This section also describes the problem statement. Section 3 presents the formulation of mathematical model, while case studies are investigated to demonstrate the result of the proposed model in Sect. 4. Finally, the chapter concludes and draws the future work in Sect. 5.

2 Problem Description and the "Bartender Concept"

The automation technology in combination with advances in production management has dramatically changed the equipment used by manufacturing companies as well as the issues in planning and control. These changes have led to an enormous increase in efficiency and flexibility so that progress in term of automation has become a necessity to survive global competition (Crama and van de Klundert 1999). As a result, highly automated or unmanned production systems have become more popular in several industrial areas such as chemical and plastics, automotive, pump manufacturing, etc. A typical automatic production system includes intelligent

and flexible machines programmed and grouped into cells in such a way that entire production of each part can be carried out within one of the cells. This machine pooling helps to reduce material handling activities. Within a manufacturing cell, material handling is usually performed by one (or several) robots/automatic guided vehicles or human operator. When the former is the case, the performance of the cell becomes highly dependent on the interaction between automatic material handling device(s) and the machine(s) (Crama and van de Klundert 1999).

The relatively small number of machines and material handling device(s) in flexible cells, along with their high degree of automation make them an ideal environment for automated production scheduling. In general, a flexible manufacturing system with multiple machines, multiple storages, and a single robot transporting parts in between machines, also between machines and storages or from storages to feed machines are taken into account. For instances, in an automated printed circuit board assembling environment, the robot has to transport the boards in-between insertion machines and also between insertion machines and buffers at the right time before the adhesive become dry (Maimon et al. 2000), or in car-parts manufacturing factory, the robot moves a fixed track to load parts into the pallets from the storage line. The set of move requests (tasks) is assumed to be known in advance, thus the robot can be considered to have a set of tasks to carry out at each planning point. The problem is to sequence the tasks, with the objective of minimizing some performance measure such as the total travel time of the robot. Apart from the total travel time of the robot, other local performance measures can be considered for this problem such as the amount of time a task waiting for to be completed by the robot. Alternatively, completion time of a task could also be modeled as a constraint to ensure that a task is completed before a certain time. An example of it is in water fabrication facilities in which time windows for the tasks to be carried out are a consideration (Maimon et al. 2000). Besides, industrial production technology has traditionally focused on improving the quality and quantity of production with little attention paid to environmental and social costs. For instance, fixed industrial robots along with safety fences, conveyors, and prescribed material flows are ideal for repetitive tasks, increase manufacturing throughput and quality. However, operating these kinds of robots have consumed a substantial amount of energy and required high costs of changeover and worker protections. The growth in environmental consciousness has led to a significant change in this attitude, and—willingly or otherwise—businesses and manufacturing are now forced to confront the consequences of their actions (Sarkis 1995). Modern technologies with automatic material handling devices (e.g. flexible mobile robots with embedded batteries, manipulation arms and various end-effectors) incorporated to production offer the promise of gaining benefits of environmental consciousness. These kinds of devices or robots might initially be more expensive than other solutions. However, through additional creations of values and by a faster adaptation to changes with new levels of robustness, availability, and completeness of jobs, the alternative solutions could yield an earlier return of investment, safer and cleaner facilities, lower future costs for disposal and worker protection, as well as improved product quality and higher productivity of entire production.

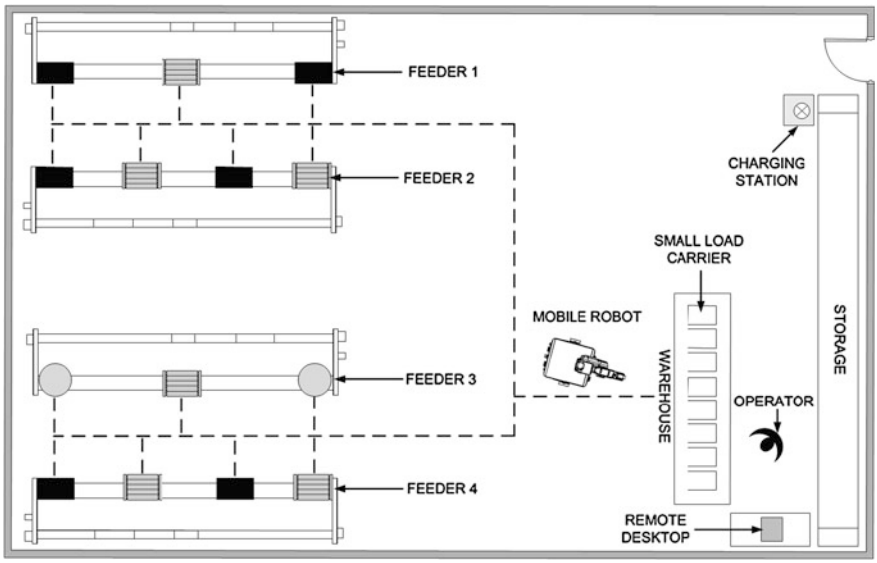

Fig. 1 Illustration of the "Bartender Concept" in the manufacturing cell

The automatic material handling devices such as mobile robots can be used in either fully automated or in-between fully automated and strictly manual production systems. Thus, they have been widely employed in not only small companies, which focus on exact application and a smaller range of products, but also large companies, which can diversify their applications in the longer term and larger range. However, the increasing market uncertainties and environmental issues require both small and large series productions to quickly respond to changing customer demands and implementing environmentally conscious manufacturing. Hence, the companies need to understand how these devices could be used to meet those requirements. Within the scope of this study, the interaction between one of the automatic material handling devices, a mobile robot, and machines in a cell, which focuses on a class of robotic scheduling, is considered under environmental consciousness. Figure 1 shows a typical layout of the manufacturing cell. In particular, the work is developed for a real cell that produced parts or components for the pump manufacturing industry at a factory in Denmark.

Before coordinating a mobile robot to industrial environment, the manufacturing cell has one or several production lines, which consists of multiple machines. Feeders are designed to automatically supply parts or components to these machines. Pallets or boxes, which contain the parts, are placed next to these feeders. Multiple-parts feeding that is the process of loading many parts at a time into feeders from the pallets or boxes is manually performed, non-value adding manufacturing task, and quite often disruptive (in-between and periodic) for production workers. In case that the workers forget to fill the feeders with parts, this may lead to a serious problem of stopping the production lines. A strategy that can reduce the dependence of the

multiple-part feeding tasks on human intervention is having a mobile robot taking over parts of what humans used to do. However, to utilize the autonomous mobile robot in this scenario requires changing the working environment in the former stage and carefully planning in the latter stage.

The "Bartender Concept" is implemented to meet the requirement of the former stage as well as to make multiple-part feeding tasks performed by the mobile robot more flexible and complete. In this concept, instead of scattered pallets or boxes containing parts next to the feeders, a warehouse (the bar) is created to gather different parts into one area. An operator (the bartender) puts parts into small load carriers (SLCs) which are placed in the warehouse. The robot retrieves and carries these SLCs, moves to each feeder and feeds all parts inside a SLC to each feeder, then returns to the warehouse to unload all empty SLCs and take filled SLCs if needed. It can be seen that the benefits obtained by applying the "Bartender Concept" include safer, cleaner facilities and better opportunities for process control.

In the latter stage, with the restructured production environment, the feeders have to be served a number of times in order to keep producing all of products, so the robot has a set of multiple-parts feeding tasks to carry out during production time of products. The number of these feeding tasks is mainly influenced by the number of parts, or the total weight of parts, inside a SLC that the robot is able to carry. The more parts, or the heavier, the robot transports each time the fewer tasks it has to perform in a given planning horizon and vice versa. In order to accomplish all the movements with the smallest consumed amount of battery energy, the total traveling time of the robot is an important objective to be considered. Hence, it is crucial to determine in which way the robot should supply the feeders of machines with parts in order to minimize its total traveling time within the manufacturing cell while preventing the production lines from stopping working. A mathematical formulation is developed to achieve this objective in the next section.

3 Mathematical Formulation

In this study, a mix-integer programming (MIP) model is developed to determine optimal route of the mobile robot visiting a number of feeders to process multiple-parts feeding tasks. The model is inspired by well-known vehicle routing problem in which all the feeders (customers) corresponding to SLCs deliveries are known in advance. The (s, Q) inventory system (Silver et al. 1998) is applied to determined time-windows for part-feeding tasks. The upper bound of time window is set on the time when the number of parts inside a feeder drops to a certain level s, while the lower bound is defined on the time when there is no part in the feeder. The task for each feeder must start within an associated time window and the robot must remain at the feeder location during service. Soft time windows can be violated, while hard time windows do not allow the robot to arrive at a feeder after the latest time to begin service. In the latter case if it arrives before the feeder is ready to begin service, it waits (Toth and Vigo 2002). We will concentrate on hard

time window scenarios in this model. The robot is based at a single warehouse and the capacity restriction of the robot is imposed. The certain number of parts Q is predefined to fill feeders where the robot visits each time.

MIP problem consists of some of decision variables that are constrained to have only integer values. Integer variables make an optimization problem non-convex, and therefore far more difficult to solve. Memory and solution time may exponentially rise as the size of the problem increases with more added integer variables. Therefore, in practice the MIP model can be applied to small-scale problems with a few numbers of feeders, products and short planning horizon. Under these scenarios, the MIP model is reasonably fast to give exact optimal solutions, which can be used as reference points to quantify the scale of benefits achieved by a meta-heuristic method further developed. Moreover, this MIP model attempt to minimize total traveling time of the robot; in other words, minimize battery energy consumption that is one of the important aspects of ecological production. Consequently, by saving battery energy for non-value added transportation of parts around shop floor, the mobile robot is able to take full advantage of flexible characteristics to perform other tasks at other workstations that leads to higher productivity of the entire production. Assumptions, notations and formulation for the MIP model are extensively described in the following subsections.

3.1 Assumptions

- A fully automatic mobile robot is taken into account in disturbance free environment.
- The robot can carry several SLCs at a time.
- All tasks are periodic.
- All tasks are assigned to the same robot.
- All tasks are independent which means that there are no precedence constraints.
- Working time and traveling time of the robot between any two locations, in which either one of the locations can be a feeder or warehouse place, are known.
- Consuming rate of parts in a feeder is known.
- All of feeders of machines have to be fed up to maximum level and the robot starts from the warehouse at the initial stage.

3.2 Notations

N: set of all task (0: index of task at the warehouse)
n_i: total number of times which task i has to be executed
R: maximum number of route $(R = \sum n_i, \forall i \in N \backslash \{0\})$
e_{ik}: k-th released time of task i
d_{ik}: k-th due time of task i

p_i: period time of task i
w_i: working time per SLC of task i
t_{ij}: traveling time of robot from task i location to task j location
c_i: consuming rate of parts in feeder at task i location
v_i: minimum level of parts in feeder at task i location
u_i: maximum level of parts in feeder at task i location
Q: maximum number of SLCs that can be carried by the robot

Decision variables :

$$x_{ik}^{jlr} = \begin{cases} 1 & \text{if robot travels from } k\text{-th task } i \text{ location to } l\text{-th task } j \text{ location in the route } r \\ 0 & \text{otherwise} \end{cases}$$

$y_{ik:}$ route number in which k-th task i belongs
$S_{ik:}$ k-th starting time of task i

3.3 Mixed-Integer Programming Model

$$\text{Objective function: } \min \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} t_{ij}x_{ik}^{jlr} \tag{1}$$

Subject to:

$$p_i = (u_i - v_i)c_i \qquad \forall i \in N\backslash\{0\} \tag{2}$$

$$e_{ik+1} = e_{ik} + p_i \qquad \forall i \in N\backslash\{0\}, k = 1 \div n_i \tag{3}$$

$$d_{ik} = e_{ik} + (v_i - 0)c_i \qquad \forall i \in N\backslash\{0\} \tag{4}$$

$$e_{ik} \le s_{ik} \le d_{ik} \qquad \forall i \in N\backslash\{0\}, k = 1 \div n_i \tag{5}$$

$$\sum_{j\in N\backslash\{0\}}\sum_{l=1}^{n_j} x_{01}^{jl1} = 1 \tag{6}$$

$$\sum_{j\in N\backslash\{0\}}\sum_{l=1}^{n_j}\sum_{r\in R} x_{01}^{jlr} \le 1 \tag{7}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i} x_{ik}^{ikr} = 0 \qquad \forall r \in R \tag{8}$$

$$\sum_{r\in R} x_{ik}^{jlr} \le |Z| - 1 \quad \forall i,j \in N, k = 1 \div n_i, l = 1 \div n_j, i \ne j, Z \subseteq Z_T, Z \ne \Phi \tag{9}$$

$$\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} x_{ik}^{jlr} = 1 \quad \forall i \in N\backslash\{0\}, k = 1 \div n_i \tag{10}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} x_{ik}^{jlr} = 1 \quad \forall j \in N\backslash\{0\}, l = 1 \div n_j \tag{11}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N\backslash\{0\}}\sum_{l=1}^{n_j} x_{ik}^{jlr} \le Q \quad \forall r \in R \tag{12}$$

$$s_{ik} + \left(w_i + t_{ij}\sum_{r\in R} x_{ik}^{jlr}\right) - L\left(1 - \sum_{r\in R} x_{ik}^{jlr}\right) + \left(y_{jl} - y_{ik}\right) \times \left(t_{i0} + w_0 + t_{0j} - t_{ij}\right) \le s_{jl} \forall i,j \in N, k = 1 \div n_i, l = 1 \div n_j, \forall r \in R \tag{13}$$

$$y_{jl} = \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} r \times x_{ik}^{jlr} \quad \forall j \in N\backslash\{0\}, l = 1 \div n_j \tag{14}$$

$$y_{jl} \ge y_{ik}\sum_{r\in R} x_{ik}^{jlr} \quad \forall i,j \in N, k = 1 \div n_i, l = 1 \div n_j \tag{15}$$

$$x_{ik}^{jlr} \in \{0,1\} \quad \forall r \in R, \forall i,j \in N, k = 1 \div n_i, l = 1 \div n_j \tag{16}$$

$$y_{ik} : \text{positive integer variable} \quad \forall i \in N, k = 1 \div n \tag{17}$$

The objective is to minimize the total traveling time of robot. Constraints (2), (3), and (4) set period time of each task, released time and due time of each execution of each task, respectively. Constraint (5) ensures that starting time of each execution of each task satisfies its time window. Constraints (6) and (7) indicate that the robot starts from the warehouse at the initial stage. Constraint (8) prevents the robot repeating an execution of one task. Constraint (9) eliminates the sub-tours among executions of tasks, where Z is a subset of Z_T, where Z_T is a set of all executions of tasks at feeders and the warehouse, and Φ denotes and empty set. Constraints (10) and (11) force each execution of each task in one route to be done exactly once. Constraint (12) forbids the robot to feed more SLCs than the maximum number of SLC Q it allows to carry. Constraint (13) handles the traveling time requirements between any pair of executions of tasks, where L is a given sufficiently large constant. In case two executions of the same task or different tasks are connected but they are not in the same route, the robot should visit the warehouse to unload empty SLCs and load filled ones. Constraint (14) assigns each execution of each task to a route and constraint (15) guarantees the ascending sequence of route numbers for executions of tasks. Constraints (16) and (17) imply the types of variables.

4 Case Studies

To examine performance of the MIP model, case studies are investigated at the CR factory at Grundfos A/S. The chosen area for the case studies is the CR 1-2-3 impeller production line that produces impellers for the CR products. The CR line consists of a warehouse and four feeders that have to be served by the mobile robot. The warehouse is indexed 0 and the feeders are indexed from 1 to 4 ($N = \{0, 1, 2, 3, 4\}$) and named Back Plate, Van Feeder 1, Van Feeder 2, and Front Plate respectively. Besides, different feeders are filled by different kinds of parts, namely back plates for feeder 1, vanes for feeder 2 and 3, front plates for feeder 4. To produce an impeller on the CR 1-2-3 line, three types of parts, which consist of six vanes (with three vans from feeder 2 and the other three vans from feeder 3), one front plate from feeder 4 and one back plate from feeder 1, are automatically assembled. The design weights of vane, front plate and back plate are 1 g, 50 g, and 50 g per part respectively. Figure 2 shows the different parts of an impeller, while Fig. 3 particularly illustrates the aforementioned production area where the presented model has been implemented. It can be seen that the safety fences and warning signs are used as described in Fig. 3 to ensure that no people enter the area as well as to prevent the mobile robot leaving that area while the implementation is taking place.

The average number of parts per SLC fed to feeder 1 or 4 is 125 (approximately 2 kg/SLC), while the average number of parts per SLC fed to feeder 2 or 3 is 1100 (approximately 1 kg/SLC). The maximum and minimum levels and consuming rate of parts in each feeder are given in Table 1, while Tables 2 and 3 show working time of robot at each task's location and traveling time of robot from one task's location to another (feeder 0 means the warehouse).

In the early design, the mobile robot has capability of carrying up to three SLCs at a time to perform multiple-part feeding tasks at the feeders. Therefore, two case studies were investigated corresponding to two maximum numbers of SLCs ($Q = 2$ and $Q = 3$) that can be carried once by the robot. These cases were executed during approximately 50 min because of robot's battery limitation. The parameters p_i, e_{ik}, and d_{ik} are respectively computed from (2), (3), and (4) in the section 3.3.

The MIP model has been coded and solved by the mathematical modeling language ILOG OPL 3.6 which is a sophisticated and computationally efficient solver that can handle linear programming problems with hundreds of thousands of variables and mixed-integer linear programming problems with tens of thousands of variables. Using this modeling language is particularly advantageous for development of new models and for documentation of models that are subjected to potential changes (Chang et al. 2001). Both case studies have 4040 decision variables including 4000 integer variables x_{ik}^{jlr}, 20 integer variables y_{ik} and 20 variables s_{ik}. The problem of the case studies have been run on a PC that has Intel Core 2 Duo CPU and 2.2 GHz processor (2 GB RAM).

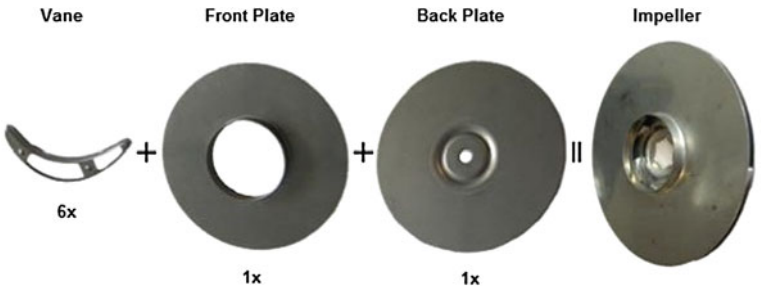

Fig. 2 Different parts of an impeller produced on the CR 1-2-3 line

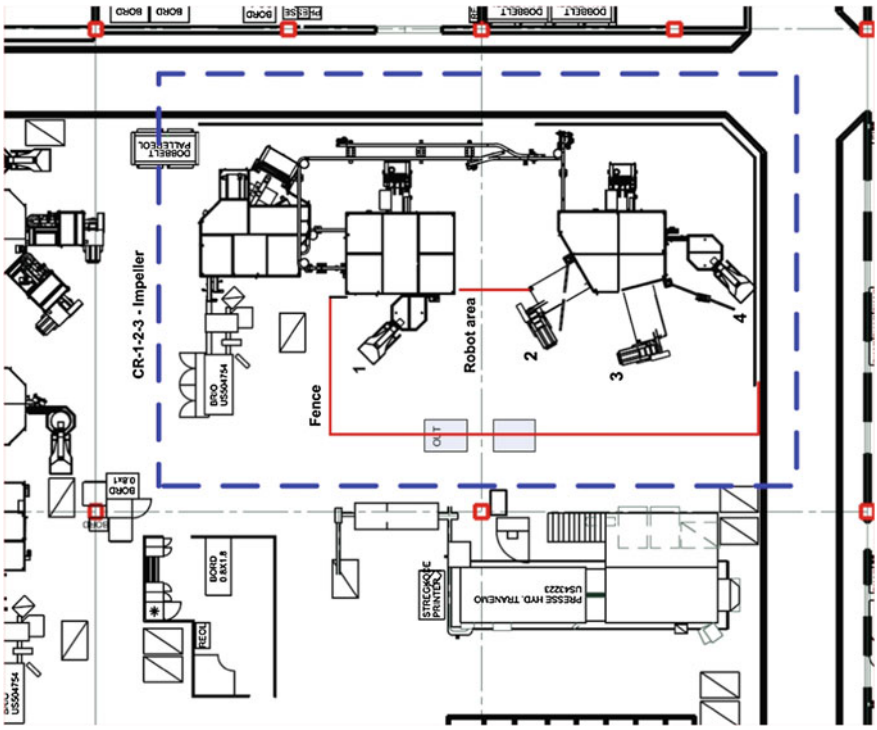

Fig. 3 CR 1-2-3 impeller production line

Table 1 Maximum and minimum levels and consuming rat of parts in feeders

Feeder	1	2	3	4
Maximum level u_i (part)	250	2000	2000	250
Minimum level v_i (part)	125	900	900	125
Consuming rate c_i (sec/part)	4.5	1.5	1.5	4.5

Table 2 Working time of robot at feeders

Feeder	0	1	2	3	4
Working time w_i (sec)	90	42	42	42	42

Table 3 Traveling time of robot from one feeder location to another

Traveling time t_{ij} (sec)	0	1	2	3	4
0	–	49	44	43	38
1	49	–	58	45	58
2	46	58	–	35	48
3	42	43	35	–	47
4	44	56	47	46	–

4.1 Case Study 1

The first case considered that the mobile robot could only carry up to two SLCs at a time during its performance. The optimal solution obtained from this case is given as $x_{01}^{411} = x_{41}^{022} = x_{02}^{112} = x_{11}^{122} = x_{12}^{033} = x_{03}^{423} = x_{42}^{433} = x_{43}^{044} = x_{04}^{134} = x_{13}^{055} = x_{05}^{445} = x_{44}^{145} = x_{14}^{066} = x_{06}^{216} = x_{21}^{316} = x_{31}^{077} = 1$, the others $x_{ik}^{jlr} = 0$ and $y_{01} = y_{41} = 1$, $y_{02} = y_{11} = y_{12} = 2$, $y_{03} = y_{42} = y_{43} = 3$, $y_{04} = y_{13} = 4$, $y_{05} = y_{44} = y_{14} = 5$, $y_{06} = y_{21} = y_{31} = 6$ which form the entire route 0 - 4 - 0 - 1 - 1 - 0 - 4 - 4 - 0 - 1 - 0 - 4 - 1 - 0 - 2 - 3 - 0, with the total traveling time being 624 s which makes up 20.8 % of the total time. The detailed solution is summarized in Table 4 and Fig. 4 below.

4.2 Case study 2

The second case considered that the mobile robot could carry up to three SLCs at a time during its performance. The optimal solution obtained from this case is given as $x_{01}^{411} = x_{41}^{111} = x_{11}^{121} = x_{12}^{022} = x_{02}^{422} = x_{42}^{432} = x_{43}^{132} = x_{13}^{033} = x_{03}^{443} = x_{44}^{044} = x_{04}^{144} = x_{14}^{314} = x_{31}^{214} = x_{21}^{055} = 1$, the others $x_{ik}^{jlr} = 0$ and $y_{01} = y_{41} = y_{11} = y_{12} = 1$, $y_{02} = y_{42} = y_{43} = y_{13} = 2$, $y_{03} = y_{44} = 3$, $y_{04} = y_{14} = y_{31} = y_{21} = 4$ which form the entire route 0 - 4 -1 - 1 - 0 - 4 - 4 - 1 - 0 - 4 - 0 - 1 - 3 - 2 - 0, with the total traveling time being 543 s which constitute of 18.1 % of the total time. The detailed solution of the case is summarized in Table 5 and Fig. 5 below.

These above optimal solutions of the case studies 1 and 2 are initial schedules for the mobile robot. The initial schedule serve as inputs to a program called Mission Planner and Control (MPC), which is implemented in VB.NET. The Mission Planner and Control program is accessed using XML-based TCP/IP communication to command and get feedbacks from the robot, and interact with ERP system of the company. By integrating the mobile robot into the general enterprise network, it is possible to plan and control globally, as the mobile robot become a resource on the same level as corresponding manufacturing device. During the practical

Table 4 Detailed optimal solution of the case study 1

Feeder/Task	Index of execution	Starting time (s_{ik})	Route
4	1	810.0	1
1	1	1125.0	2
1	2	1378.5	2
4	2	1687.5	3
4	3	1935.0	3
1	3	2250.0	4
4	4	2510.0	5
1	4	2608.0	5
2	1	2923.0	6
3	1	3000.0	6

multiple-parts feeding operations at CR 1-2-3 impeller production line, the mobile robot was able to continuously pick/place SLCs from/to the warehouse and empty them into the different feeders so that the initial schedules were executed in sequence and they prevented all of feeders running out of parts. Hence, the CR 1-2-3 production line can keep manufacturing impellers without shortage of parts that are fed from feeders.

From those case studies, it reveals that the higher maximum number of SLCs the robot can carry, the less traveling time it has; in other words, the less battery energy it consumes. It can be also observed that the robot only visits the central warehouse 3 times in the second case as opposed to 5 times in the first one. Because of fewer times on visiting the warehouse, the robot can also save its battery energy from the work consisting of unloading all empty SLCs and loading newly filled SLCs there. Consequently, when the mobile robot gains less energy usage, it can perform multiple-part feeding tasks in a longer planning horizon, or move to other workstations with its ability to automatically change various end-effectors in order to serve other types of tasks, for examples single part feeding, pre-assembly, machine tending, quality inspection, process execution during its idle time if required. These apparently lead to more sustainable production, higher productivity, and demonstrate the superior characteristics of mobile robots to industrial robots, which are attached to a fixed surface, or automatic guided vehicles, which follow markers, reflectors, guided wires on the floor in the context of environmentally conscious manufacturing.

5 Conclusions and Further Research

In this chapter, a new problem of scheduling a single mobile robot, which is incorporated into production lines considering environmental consciousness, to perform multiple-parts feeding tasks is studied. In order to reduce the human interventions and intensively utilize the mobile robot, the "Bartender Concept" was implemented to restructure the working environment. Benefits obtained from that implementation

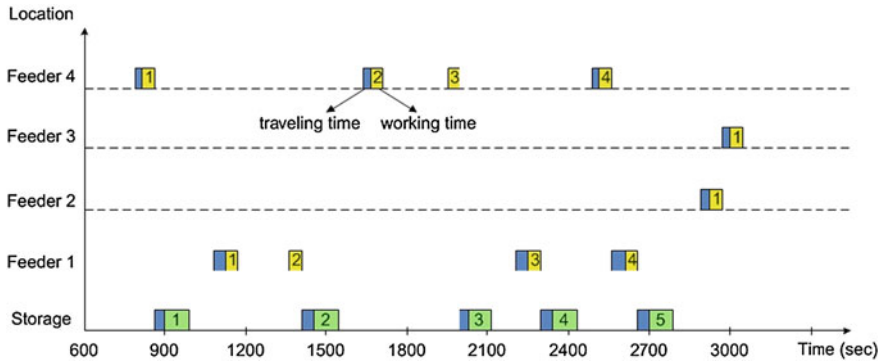

Fig. 4 Gantt chart for the optimal solution of the case study 1

Table 5 Detailed optimal solution of the case study 2

Feeder/Task	Index of execution	Starting time (s_{ik})	Route
4	1	1027.0	1
1	1	1125.0	1
1	2	1378.5	1
4	2	1687.5	2
4	3	2152.0	2
1	3	2250.0	2
4	4	2497.5	3
1	4	2812.5	4
3	1	2923.0	4
2	1	3000.0	4

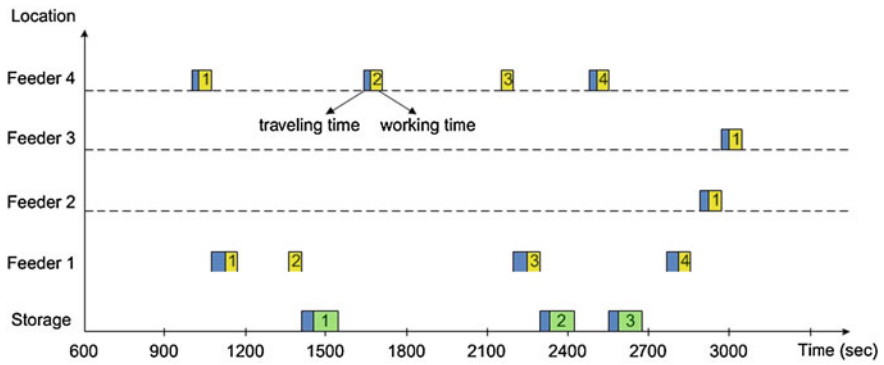

Fig. 5 Gantt chart of the optimal solution of the case study 2

consist of safer, cleaner facility layout, and better opportunities for process control. Besides, to accomplish all the tasks within allowable limit of battery capacity, it is important for mission planners to determine optimal feeding sequence to minimize

total traveling time of the mobile robot while taking into account specific features of the robot and a number of technological constraints.

A new mix-integer programming model was developed to find optimal solution for the problem. This model could be coded and solved by using the mathematical modeling language ILOG OPL 3.6 which is advantageous for development of new scenarios due to potential changes of dynamic production environment. The proposed model can be applied in practice with small-scale problems with a few numbers of feeders, products and short planning horizon. The particular real cases of the impeller production line composing of four feeders were described to show result of the proposed model. The result was quite properly applied during practical feeding operations and it demonstrated that all feeders had no shortage of parts fed to the production line. Furthermore, it can be seen from the real cases that along with the flexible characteristic of the mobile robot, if it has higher ability to carry more number of SLCs at a time, it can save time on transporting parts around the shop floor and working at the central warehouse. Consequently, this can lead to eco-efficiency gains such as less energy usage, more sustainable production and higher productivity, as well as demonstrate the superior characteristics of mobile robots to fixed industrial robots or automatic guided vehicles.

For further research, the complexity of the problem categorized as being NP-hard will increase when considering a large number of feeders and/or long planning horizon. Hence, a meta-heuristic method will be taken into account for solving the large-scale problem of mobile robot scheduling. Besides, during the implementation of the real case studies at the shop floor, different interruptions occasionally happened because of various reasons such as losing communication between the robot and the MPC program, failure of the robot to picking/placing SLCs from/to the central warehouse or empty SLCs at the feeders, even stopping the production line due to scrap materials. These interruptions require the MPC program to have the robust ability to recover the system state before errors and then re-schedule the remaining tasks based on obtained schedules and feedback from the shop floor. Hence, a re-scheduling mechanism with an integrated real-time discrete event simulation model might be developed to deal with these aforementioned real-time disturbances. Moreover, multi-objectives function concerning environmental impacts to improve eco-production and logistics could be considered for future work.

Acknowledgments This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

References

Azzone G, Noci G (1998) Identifying effective PMSs for the deployment of "Green" manufacturing strategies. Int J Oper Prod Manag 18(4):308–335

Chang GW, Aganagic M, Waight JG, Medina J, Burton T, Reeves S, Christoforidis M (2001) Experiences with mixed Integer linear programming based approaches on short-term hydro scheduling. IEEE Transac Power Syst 16(4):743–749

Chen FF, Su Q (1995) Scheduling single-gripper Gantry Robots in tightly coupled serial production lines: Optimum vs. Push/Pull concept based sequences. J Manuf Syst 14(3):139–147
Crama Y, van de Klundert J (1997) Cyclic scheduling of identical parts in a robotic cell. Oper Res 45(6):952–965
Crama Y, van de Klundert J (1999) Cyclic scheduling in 3-machine robotic flow shops. J Sched 2(1):35–54
Crama Y, Kats V, van de Klundert J, Levner E (2000) Cyclic scheduling in robotic flow shops. Annals Oper Res 96:97–124
Darnall N, Nehmann G, Priest J, Sarkis J (1994) A review of environmentally conscious manufacturing theory and practices. Int J Environ Conscious Des Manuf 3(2):49–58
Dror M, Stulman A (1987) Optimizing Robot's service movement: a one dimensional case. Comput Ind Eng 12(1):39–46
Gungor A, Gupta SM (1999) Issues in environmentally conscious manufacturing and product recovery: a survey. Comput Ind Eng 36(4):811–853
Ilgin MA, Gupta SM (2010) Environmentally conscious manufacturing and product recovery (ECMPRO): a review of the State of the Art. J Environ Manag 91(3):563–591
Jin K, Balasubramaniam PA (2003) Fuzzy model for environmental benign process planning selection. 3rd International Symposium on Environmentally Conscious Design and Inverse Manufacturing, pp. 731–732, doi: 10.1109/VETECF.2003.240467
Kats V, Levner E (2010) Parametric algorithms for 2-cyclic robot scheduling with interval processing times. J Sched 14(3):267–279
Kats V, Levner E (2011) A faster algorithm for 2-cyclic robotic scheduling with a fixed robot route and interval processing times. Eur J Oper Res 209(1):51–56
Krishnan N, Sheng PS (2000) Environmental versus conventional planning for machined components. CIRP Annals Manuf Technol 49(1):363–366
Maimon O, Braha D, Seth V (2000) A neural network approach for a robot task sequencing problem. Artif Intell Eng 14(2):175–189
Nielsen I, Lim M, Nielsen P (2010) Optimizing Supply Chain Waste Management through the Use of RFID Technology. 2010 IEEE International Conference on RFID-Technology and Applications, pp. 296–30, doi: 10.1109/RFID-TA.2010.5529921
Sarkis J (1995) Manufacturing strategy and environmental consciousness. Technovation 15(2):79–97
Sarkis J, Rasheed A (1995) Greening the manufacturing function. Bus Horiz 38(5):17–27
Silver EA, Pyke DF, Peterson R (eds) (1998) Inventory management and production planning and scheduling. Wiley, New York
Srinivasan M, Sheng P (1999a) Feature based process planning in environmentally conscious machining—Part 1: Microplanning. Robotics Comput Integ Manuf 15(3):257–270
Srinivasan M, Sheng P (1999b) Feature based process planning in environmentally conscious machining—Part 2: Macroplanning. Robotics Comput Integ Manuf 15(3):271–281
Suárez R, Rosell J (2005) Feeding sequence selection in a manufacturing cell with four parallel machines. Robotics Comput Integ Manuf 21(3):185–195
Toth P, Vigo D (eds) (2002) The vehicle routing problem. SIAM, Philadelphia

Paper D

A genetic algorithm-based heuristic for part-feeding mobile robot scheduling problem

This paper has been published in Trends in PAAMS, AISC 157. Springer-Verlag Berlin Heidelberg, pp. 85–92

A Genetic Algorithm-Based Heuristic for Part-Feeding Mobile Robot Scheduling Problem

Quang-Vinh Dang, Izabela Ewa Nielsen, and Grzegorz Bocewicz

Abstract. This present study deals with the problem of sequencing feeding tasks of a single mobile robot with manipulation arm which is able to provide parts or components for feeders of machines in a manufacturing cell. The mobile robot has to be scheduled in order to keep machines within the cell producing products without any shortage of parts. A method based on the characteristics of feeders and inspired by the (s, Q) inventory system, is thus applied to define time windows for feeding tasks of the robot. The performance criterion is to minimize total traveling time of the robot in a given planning horizon. A genetic algorithm-based heuristic is developed to find the near optimal solution for the problem. A case study is implemented at an impeller production line in a factory to demonstrate the result of the proposed approach.

Keywords: Scheduling, Mobile Robot, Genetic Algorithm, Part Feeding.

1 Introduction

Today's production systems range from fully automated to strictly manual. While the former is very efficient in high volumes but less flexible, the latter is reversed. Therefore, manufactures visualize the need for transformable production systems that combines the best of both worlds by using new assistive automation and mobile robots. A given problem is particularly considered for mobile robots with manipulation arms which will automate extended logistic tasks by not only transporting but also collecting containers of parts and emptying them into the place

Quang-Vinh Dang · Izabela Ewa Nielsen
Dept. of Mechanical and Manufacturing Engineering, Aalborg University, Denmark
e-mail: {vinhise,izabela}@m-tech.aau.dk

Grzegorz Bocewicz
Dept. of Computer Science and Management, Koszalin University of Technology, Poland
e-mail: bocewicz@ie.tu.koszalin.pl

J.M.C. Rodríguez et al. (Eds.): Trends in PAAMS, AISC 157, pp. 85–92.
springerlink.com

needed. In that context mobile robots play the role of agents [12], attempting to reach their goals while following rules specific for a given production system. So, the considered systems are treated as multi-agent ones in which each robot can be seen as an autonomous object capable to undertake decisions about moving, feeding, emptying containers and completing operations, etc.

Feeding operation studied in this paper is a kind of extended logistic tasks. However, to utilize agents in an efficient manner requires the ability to properly schedule these feeding tasks. Hence, it is important to plan in which sequence agents' process feeding operations so that they could effectively work while satisfying a number of technological constraints.

Robot scheduling problem has attracted interest of researchers in recent decades. Crama and van de Klundert [1] considered the flow shop problem with one transporting robot and one type of product to find shortest cyclic schedule for the robot. Afterwards, they demonstrated that the sequence of activities whose execution produces one part yields optimal production rates for three-machine robotic flow shops [2]. Crame et al. [3] also presented a survey of cyclic robotic scheduling problem along with their existing solution approaches. Dror and Stulman [5] dealt with the problem of optimizing one-dimensional robot's service movements. Kats and Levner [7, 8] considered m-machine production line processing identical parts served by a mobile robot to find the minimum cycle time for 2-cyclic schedules. Maimon et al. [9] introduced a neural network method for a material-handling robot task-sequencing problem. Suárez and Rosell [11] built several strategies and simulation model to deal with the real case of feeding sequence selection in a manufacturing cell consisting of four identical machines. Most of the work and theory foundation considered approaches for scheduling robots which are usually inflexible, move only on fixed path and repeatedly perform a limited sequence of activities. There is still lack of approaches for scheduling mobile robots which are able to move around within a manufacturing cell to process extended logistic tasks with specific time windows and limitation in carrying capacity of mobile robots. Such a problem is modeled in several respects comparable to Asymmetric Traveling Salesman Problem (ATSP) in which the traveling time/cost might be different in two directions of a path. In addition, the ATSP belongs to NP-complete class [8] in which the required computational time exponentially grows with the size of the problem. Therefore, in this paper we focus on developing a computationally efficient heuristic, namely genetic algorithm-based heuristic for scheduling of mobile robot for feeding tasks which could be predetermined based on characteristics of feeders.

The remainder of this paper is organized as follows: in the next section, problem description is described while a genetic algorithm-based heuristic is presented in Section 3. A case study is investigated to demonstrate the result of the proposed algorithm in Section 4. Finally, conclusions are drawn in Section 5.

2 Problem Description

The work is developed for a real cell that produces parts for the pump manufacturing industry at a factory in Denmark. The manufacturing cell consists of a central

warehouse, a single mobile robot (an agent), multiple machines which form production lines and feeders which are designed to automatically supply parts to these machines. The robot carries one or several small load carriers (SLCs) containing parts from the warehouse, moves to feeder locations, empty all parts inside SLCs to feeders, then returns to the warehouse so as to unload all empty SLCs and take filled SLCs. Each feeder (or task), which possesses its own characteristics such as maximum, minimum levels and consuming rate of parts, has to be served a number of times in order to keep manufacturing products, so the robot has a set of sub-tasks possessing time windows to carry out on each feeder during planning horizon. The manufacturing cell considered can be seen as an agent system that can be easily extended to the multi-agent one.

To enable the construction of a feeding schedule for the mobile robot, the following assumptions are made: a fully automatic mobile robot is considered in disturbance free environment; the robot can carry several SLCs at a time with limitation in payload; all tasks are periodic, independent, and assigned to the same robot; working time and traveling time of the robot between any two locations, in which either one of the locations could be a feeder or warehouse, are known; consuming rate of parts in a feeder is known; all feeders of machines have to be fed up to maximum level and the robot starts from the warehouse at the initial stage.

In order to accomplish all the movements with a smallest consumed amount of battery energy, the total traveling time of the robot is an important objective to be considered. Concerning computational time of the problem belonging to NP-complete class, it exponentially grows with the size of the problem (i.e. longer planning horizon, larger number of feeders). It is therefore necessary to develop a computationally effective algorithm that determines in which way the robot should supply the feeders with parts in order to minimize its total traveling time within the manufacturing cell while satisfying a number of technological constraints.

3 Genetic Algorithm-Based Heuristic

Among many meta-heuristics, genetic algorithm, a well-known method, is applied to develop a heuristic, shown in Figure 1, which is allowed to convert the aforementioned problem to the way that a near optimal solution could be found. The genetic algorithm-based heuristic consists of the following steps: (i) genetic representation and initialization, (ii) adjustment mechanism and fitness evaluation, (iii) selection, and (iv) crossover and mutation.

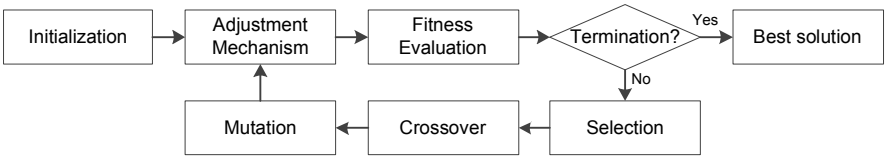

Fig. 1 Flow chart of the genetic algorithm-based heuristic

(i) Genetic Representation and Initialization

For the problem under consideration, a solution can be represented by a chromosome as shown in Figure 2. Each gene in a chromosome consists of two parts. The first part is the index of a task (feeder) while the second one implies the index of sub-task of that task. The length of each chromosome is total number of sub-tasks of all tasks which the mobile robot has to perform during the planning horizon.

1,2	2,1	1,1	3,1	...	*i,k*	...	*n,m*

Fig. 2 Genetic representation

For initial generation, genes on a chromosome are randomly filled with sub-tasks of all tasks until the end of length of that chromosome.

(ii) Adjustment Mechanism and Fitness Evaluation

After initialization or crossover and mutation operations, chromosomes are adjusted to be valid and then calculate their fitness values. A valid chromosome should satisfy two constraints about time windows of sub-tasks and capacity Q of the robot. For the first constraint, the start time of a sub-task of a task should be in-between release time and due time of that sub-task which could be determined by maximum level, minimum level and consuming rates of parts, while the second constraint requires the robot not to serve number of sub-tasks greater than number of SLCs it is carrying. An adjustment mechanism as below is applied to each chromosome in the initial generation or descendant so as to take these constraints into account.

Step 1: For each task, rearrange its sub-tasks in ascending order of their indices
Step 2: Rearrange considering time windows of all sub-tasks
- *Step 2.1:* Compute start time of the current sub-task. If start time of the current sub-task satisfies its time window then move to the next sub-task; otherwise go to step 2.2
- *Step 2.2:* Considering from the first sub-task to the current sub-task, make a list of candidates of sub-tasks whose release times are greater than that of the current one
- *Step 2.3:* Select randomly a candidate from the list
- *Step 2.*4: Insert the current sub-task to the position of the selected candidate
- *Step 2.*5: Re-compute start times of the sub-tasks from the position of the selected candidate. If all start times of the sub-tasks satisfy their time windows then go to step 3; otherwise go back to Step 2.3. If none of candidate is selected then discard this chromosome, generate a new one instead and go back to Step 1

Step 3: Rearrange considering capacity of the robot and time windows

Step 3.1: From the latest sub-task at the warehouse after every Q sub-tasks, add another sub-task at the warehouse of the robot.

Step 3.2: Re-compute start times of the next Q sub-tasks from the newly added sub-task at the warehouse. If all start times of Q sub-tasks satisfy their time windows then go back to Step 3.1; otherwise go to Step 3.3

Step 3.3: $Q = Q - 1$ and go back to Step 3.1. If $Q = 0$ then discard this chromosome, generate a new one instead and go back to Step 1. Note that the decrease of Q occurs only in the current loop, it will turn back to its predetermined value in the new loop.

Following the adjustment mechanism, the fitness evaluation will be taken place. The fitness value of a chromosome equals the total traveling time which the mobile robot moves from location of the first sub-task to location of the last one.

(iii) Selection

Various evolutionary methods can be applied to this problem. $(\mu + \lambda)$ selection is used to choose chromosomes for reproduction. Under this method, μ parents and λ offspring compete for survival and the μ best out of the offspring and old parents, in other words, the μ lowest in terms of the total traveling time, are selected as the parents of the next generation. Such selection mechanism guarantees that the best solutions up to now are always in the parent generation.

(iv) Crossover and Mutation

Crossover and mutation are main genetic operators. A crossover generates offspring by combining the information contained in the chromosomes of parents so that new chromosomes will have the good features of the parents' chromosomes. The Roulette-wheel selection is used in the algorithm, which probabilistically selects chromosomes of parents based on their fitness values (Goldberg, 1989). Genes on the selected chromosomes, which represent sub-tasks at the warehouse, are removed before recombination. An offspring then is generated with order crossover (OX) described as follow. Two cut points to be randomly chosen on the parent chromosomes. The string between these cut points in the first parent are first copied to the offspring. The remaining positions are filled by considering the sequence of genes in the second parent starting after the second cut point (when reaching to the end of chromosome, the sequence continues at position 1) [10]. The order crossover acts with probability P_c. After crossover, some offspring undergo mutation operator which is applied with probability P_m. The operation of mutation selects two positions within a chromosome at random and then inverts the substring between these two positions to produce heterogeneous chromosomes to avoid premature convergence of the algorithm.

4 Case Study

To examine performance of the proposed algorithm, a case study is investigated at the CR factory at Grundfos A/S. The chosen area for this case study is the CR 1-2-3 impeller production line which produces impellers for the CR products. The CR line consists of four feeders that have to be served by the mobile robot. Besides, different feeders are filled by different kinds of parts, namely back plates for feeder 1, vanes for feeder 2 and 3, front plates for feeder 4. On the CR line, a number of vanes are welded together with back and front plates to produce an impeller. Fig. 3 below particularly illustrates the aforementioned production area.

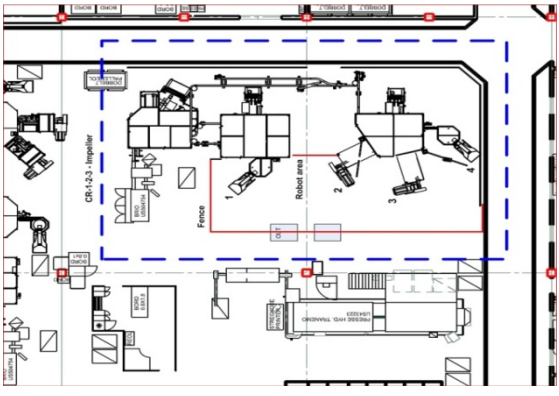

Fig. 3 CR 1-2-3 impeller production line

The maximum number of SLCs carried by the robot is 2. The average number of parts per SLC fed to feeder 1 or 4 is 125 (approximately 2 kg/SLC), while the average number of parts per SLC fed to feeder 2 or 3 is 1100 (approximately 1 kg/SLC). The maximum, minimum levels, consuming rate of parts and working time of robot at feeders are given in Table 1, while Tables 2 shows traveling time of robot from one location of feeder to another (feeder 0 means the warehouse).

Table 1 Maximum, minimum levels, consuming rate, and working time of robot at feeders

Feeder/Task	0	1	2	3	4
Maximum level (part)	-	250	2000	2000	250
Minimum level (part)	-	125	900	900	125
Consuming rate (sec/part)	-	4.5	1.5	1.5	4.5
Working time of robot (sec)	90	42	42	42	42

Table 2 Traveling time of robot from one location to another

Traveling time (sec)	0	1	2	3	4
0	-	49	44	43	38
1	49	-	58	45	58
2	46	58	-	35	48
3	42	43	35	-	47
4	44	56	47	46	-

The case study is investigated during 45 minutes because of battery limitation of the mobile robot. The proposed heuristic has been programmed in VB.NET and run on a PC that has Core i5 CPU, 2.67 GHz processor, and 4 GB RAM. A population size of 20 is used and probabilities of crossover and mutation are set to be 0.4 and 0.1, respectively. The termination is to stop at the generation of 100 or if no improvement is made after 50 generation. Figure 4 shows the convergence of the best solution of the proposed heuristic.

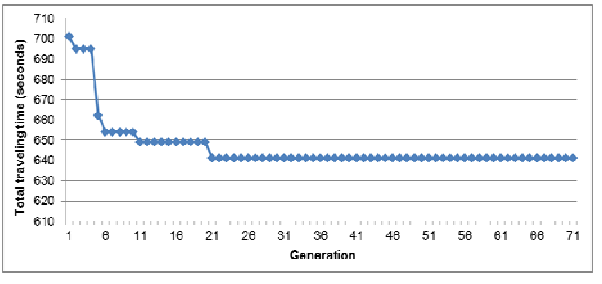

Fig. 4 Convergence of the best solution

The best solution obtained is given as: 0 - 4 - 1 - 0 - 4 - 1 - 0 - 4 - 3 - 0 - 1 - 1 - 0 - 4 - 2 - 0, with total traveling time being 641 seconds which makes up 23.7% of the planning horizon and the computational time of this case is less than a second. The proposed heuristic is significantly fast to find the near optimal solution in comparison with the optimal solution of 624 seconds and the computational time of 184 seconds which were obtained by the MIP model developed by Dang [4]. The Gantt chart of the best solution is shown in Fig. 5 below.

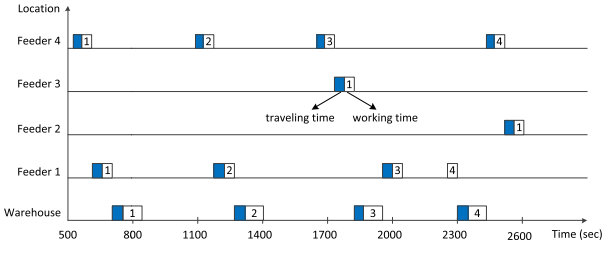

Fig. 5 Gantt chart for the best solution of the case study

5 Conclusions

In this paper, a new problem of scheduling a single mobile robot which performs part-feeding tasks is considered. To accomplish all tasks in the planning horizon within allowable limit of battery capacity, it is important to determine feeding sequences which minimize total traveling time of the mobile robot while taking into account a number of practical constraints. A genetic algorithm-based heuristic was developed to find near optimal solutions for the problem. The particular real case of an impeller production line was described to demonstrate the result of the proposed heuristic. The result showed that the proposed heuristic was significantly fast to obtain the near optimal solution. The solution is useful to managers for decision making at an operational level and the approach could be also applied in variety of tasks of not only mobile robots but also automatic guided vehicles.

The presented approach provides a solid framework that enables to model and evaluate scheduling tasks of multi-agent systems. For further research, re-scheduling mechanisms based on obtained schedules and feedback from the mobile robot fleet and shop floor should be developed to deal with real-time disturbances such as broken-down machines or unexpected shortage of parts of feeders.

Acknowledgments. "This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS".

References

[1] Crama, Y., van de Klundert, J.: Cyclic scheduling of identical parts in a robotic cell. Oper. Res. 45, 952–965 (1997)
[2] Crama, Y., van de Klundert, J.: Cyclic scheduling in 3-machine robotic flow shops. J. Sched. 2, 35–54 (1999)
[3] Crama, Y., Kats, V., van de Klundert, J., Levner, E.: Cyclic scheduling in robotic flow shops. Ann. Oper. Res. 96, 97–124 (2000)
[4] Dang, Q.V., Nielsen, I.E., Steger-Jensen, K.: Scheduling a single mobile robot for feeding tasks in a manufacturing cell. In: Proc. of Int. Conf. Adv. Prod. Manag. Syst., Norway (2011)
[5] Dror, M., Stulman, A.: Optimizing robot's service movement: a one dimensional case. Comput. Ind. Eng. 12, 39–46 (1987)
[6] Goldberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, New York (1989)
[7] Kats, V., Levner, E.: Parametric algorithms for 2-cyclic robot scheduling with interval processing times. J. Sched. (2010), doi:10.1007/s10951-010-0166-0
[8] Kats, V., Levner, E.: A faster algorithm for 2-cyclic robotic scheduling with a fixed robot route and interval processing times. Eur. J. Oper. Res. 209, 51–56 (2011)
[9] Maimon, O., Braha, D., Seth, V.: A neural network approach for a robot task sequencing problem. Artif. Intell. Eng. 14, 175–189 (2000)
[10] Potvin, J.Y.: Genetic algorithms for traveling salesman problem. Ann. Oper. Res. 63, 339–370 (1996)
[11] Suárez, R., Rosell, J.: Feeding sequence selection in a manufacturing cell with four parallel machines. Robot Comput. Integrated Manuf. 21, 185–195 (2005)
[12] Shyu, J.-H., Liu, A., Kao-Shing, H.: A multi-agent architecture for mobile robot navigation control. In: Proceedings Tenth IEEE International Conference on Tools with Artificial Intelligence, pp. 50–57 (1998)

Paper E

Scheduling a single mobile robot for part-feeding tasks of production lines

This paper has been published in Journal of Intelligent Manufacturing. Online First. DOI: 10.1007/s10845-013-0729-y

J Intell Manuf
DOI 10.1007/s10845-013-0729-y

Scheduling a single mobile robot for part-feeding tasks of production lines

Quang-Vinh Dang · Izabela Nielsen · Kenn Steger-Jensen · Ole Madsen

Received: 10 May 2012 / Accepted: 2 January 2013

Abstract This study deals with the problem of sequencing feeding tasks of a single mobile robot which is able to provide parts for feeders of machines on production lines. The mobile robot has to be scheduled in order to stoppage from lack of parts in the production line. A method based on the characteristics of feeders and inspired by the (s, Q) inventory system, is thus applied to define time windows for the feeding tasks of the robot. The capacity of the robot is also taken into consideration. The performance criterion is to minimize total traveling time of the robot for a given planning horizon. A genetic algorithm-based heuristics is presented which results in a significant increase in the speed of finding near-optimal solutions. To evaluate the performance of the genetic algorithm-based heuristic, a mixed-integer programming model has been developed for the problem. A case study is implemented at an impeller production line in a real factory and computational experiments are also conducted to demonstrate the effectiveness of the proposed approach.

Keywords Scheduling · Mobile robot · Genetic algorithm · MIP · Part feeding

Introduction

Today's production systems range from fully automated to strictly manual. While the former is very efficient in high volumes but less flexible, the latter has the opposite characteristics. However, manufacturers express a need for transformable production systems that combines the best of both worlds by using new assistive automation and mobile robots. Mobile robots have the capability to move around within their environments and are not fixed to one physical location. With embedded batteries and manipulation arms, mobile robots are more flexible and able to perform various tasks at different workstations of production lines. The tasks include such processes as: transporting and feeding materials, machine tending, pre-assembly or quality inspection. Moreover, using mobile robots can lead to production efficiency gains such as less energy usage or lower tool-changing costs than typical industrial robots fixed to one location. The superior abilities of these robots pave the way for meeting the needs of the aforementioned new production systems. In this paper, a particular problem is considered. The problem consists of a single mobile robot with a manipulation arm which will perform part-feeding tasks by collecting and transporting containers of parts and emptying them into feeders where needed. To utilize the mobile robot in an efficient manner requires the ability to properly schedule these part-feeding tasks in relation to the needs of a given production line. Hence, it is important to plan in which sequences the mobile robot processes part-feeding tasks so that it could effectively work while satisfying a number of technological constraints.

The problem of scheduling a single mobile robot for part-feeding tasks has previously been modeled as a variant of the Asymmetric Traveling Salesman Problem (ATSP). The ATSP belongs to the class of NP-hard combinatorial optimization problem (Germs et al. 2012). The main novelty of this research is that the problem simultaneously considers hard time windows of tasks and the limitation of carrying capacity on the single mobile robot. Because of these constraints, the robot has to serve a set of feeders in more than one route during a given planning horizon while still meeting the time window constraints. This means that the number of routes and the sequence which the robot travels these, has to be determined in order to minimize its total

Q.-V. Dang · I. Nielsen (✉) · K. Steger-Jensen · O. Madsen
Department of Mechanical and Manufacturing Engineering, Aalborg University, Fibigerstræde 16, 9220 Aalborg, Denmark
e-mail: izabela@m-tech.aau.dk

Published online: 17 January 2013

traveling time. Note that making decision in which sequence the robot should perform tasks is part of real-time operations. This gives the added requirement that the best sequences of tasks must be obtained quickly. Moreover, the complexity of the problem rapidly rises when the robot has to serve more feeders and/or work in a longer planning horizon. Therefore, in this paper focus is on developing a computationally efficient approach, namely GA-based heuristic for scheduling part-feeding tasks of the mobile robot. The time windows of these part-feeding tasks are determined based on characteristics of the feeders and the (s, Q) inventory system (Silver et al. 1998). To evaluate the performance of the proposed heuristic, a mixed-integer programming (MIP) model is also presented.

The remainder of this paper is organized as follows. In the next section, the theoretical foundation of the research is described. The problem description is presented in third section and the mathematical model of the problem is formulated in fourth section. The solution methodology for the formulation using a GA-based heuristic is discussed in fifth section. Sixth section illustrates the results of a case study, using real-world data from a Danish production company, from the proposed heuristic and compares its performance with that of the MIP model. Computational experiments are also conducted in this section. Finally, conclusions and future research directions are drawn in last section.

Theoretical foundations

The problem of scheduling part-feeding tasks of the mobile robot has been modeled in several respects comparable to the Traveling Salesman Problem (TSP), but it is different from the typical TSP in the sense that the traveling time/cost might be different in two directions of a path. The problem hence is similar to but not identical to the ATSP because of some additional constraints which the problem possesses. Several approaches and models for exact or heuristic algorithms have been proposed to address problems of this type. Carpaneto and Toth (1980) present a branch-and-bound algorithm for the ATSP based on the sub-tour elimination approach or the Hungarian algorithm. They discuss a new selection procedure for the sub-tour to be split and the ordering of the arcs in the selected sub-tour. A similar approach is presented by Syslo et al. (1983). This approach is based on the Hungarian algorithm and reduces the original distance matrix till an optimal solution is obtained. It is shown that the execution time is strongly dependent on the problem instance and increases with the size of the network. Miller and Pekny (1991) survey methods such as branch-and-bound and several heuristics for solving large TSP problems. A branch-and-bound algorithm is presented with computational results and is found to perform well for some classes of problems. The branch-and-bound methods (Carpaneto et al. 1995) for the ATSP use the assignment problem as a relaxation. The effectiveness of the methods derives from reduction procedures and parametric solution of the relaxed problems associated with the nodes of the branch decision tree. Turkensteen et al. (2008) introduce a branch-and-bound algorithm for the ATSP using the upper tolerances values of arcs in the corresponding assignment problem instance to determine which arcs should be excluded. The class of tolerance-based algorithms is better in solving difficult instances than the algorithm presented in e.g. Carpaneto et al. (1995). Germs et al. (2012) then enhance this approach by incorporating lower tolerances, corresponding to additional costs of a solution with a connecting arc, into the branch-and-bound search process. Ascheuer et al. (1993) present a cutting plane approach to the sequential ordering problem which is similar to the robot task-sequencing problem and finds minimum cost paths subject to precedence constraints. They outline a Linear Programming framework and discuss polynomial time separation algorithms for obtaining the solutions. The problem of order-picking in a rectangular warehouse of Automated Storage and Retrieval System is addressed by Ratliff and Rosenthal (1983). It is shown to be a solvable case of the TSP and they present an algorithm for picking an order in minimum time. Edan et al. (1991) present a near-minimum task-planning algorithm for a fruit harvesting robot to find near-optimal-time path between the N given fruit locations. The sequence of motions for the harvesting robot is obtained by solving the TSP using the geodesic distance. Dang et al. (2011) propose an MIP model to obtain the optimal feeding sequence of a mobile robot in a manufacturing cell. However, the performance of the MIP model is not evaluated and compared with other methods. For small task-scheduling problems, the aforementioned techniques can be used to find the optimal solutions of the problems. Nevertheless, they tend to get computationally intractable for large and complex problems (Maimon et al. 2000).

Larger problems call for heuristic solutions (Maimon et al. 2000). The heuristic approaches that are frequently applied to robot task-scheduling problems include: the nearest-neighbor rule in which the robot travels to the nearest pickup point from its current position (Han et al. 1987), the closest insertion algorithm which causes the smallest increase in the length of the sequence (Askin and Stanridge 1993), and dispatching rules (e.g. First-Come-First-Served) which serve the tasks in chronological order (Suárez and Rosell 2005). The above listed heuristics have shown good results and are computationally fast. Another set of heuristics including; ant colony optimization, simulated annealing, tabu search, neural network and genetic algorithm has been used to solve combinatorial optimization problem such as TSP, ATSP or robot task-scheduling problems. Tsai et al. (2004) discuss using ant colony system to solve the TSP, while López-Ibáñez and Blum (2010) apply the ant colony optimization technique to

deal with the TSP with time windows (TSPTW). Ohlmann and Thomas (2007) describe a variant of simulated annealing incorporating a variable penalty method to solve the TSPTW, while Geng et al. (2011) deal with the TSP based on an adaptive simulated annealing with greedy search. Carlton and Barnes (1996) present a robust tabu search approach to the TSPTW. Landrieu et al. (2001) presents tabu search and probabilistic tabu search for single vehicle pickup and delivery problem. Hurink and Knust (2002) propose a tabu search algorithm for scheduling a single robot in a job-shop environment. Hasegawa et al. (2002) develop two types of searching methods based on tabu search and neural networks for solving large scale TSPs. Maimon et al. (2000) present a neural network approach successfully implementing the robot task-sequencing problem. Among the heuristics proposed in previous work, genetic algorithms (GA) have been widely used in solving TSP or ATSP because GAs have a global search ability and can easily be implemented (Chen and Chien 2011). Chatterjee et al. (1996) propose a GA with an asexual reproduction plan through a generalized mutation operator that can be applied to TSP. Moon et al. (2002) present an efficient GA with a topological sort and a new crossover operation to solve the TSP with precedence constraints. Snyder and Daskin (2006) combine a GA with a local tour improvement heuristic and encoded solution using random keys for solving the generalized TSP. Liu and Zeng (2009) present an improved GA with reinforcement mutation to solve the TSP. Choi et al. (2003) propose a GA that extends the search space by purposefully generating and including infeasible solutions to solve ATSP. Xing et al. (2008) present a novel hybrid approach incorporating a GA improved on both crossover and mutation operators and some optimization strategies such as immigration, local and global optimization for the ATSP. Chen and Tseng (1996) and Zacharia and Aspragathos (2005) introduce methods based on GAs and innovative encoding to determine the optimal sequence of robot's task points which is considered as an extension to the TSP. Dang et al. (2012b) present the bartender concept to incorporate a mobile robot into production and discuss the problem of scheduling the mobile robot under consideration of environmental consciousness concerning the usage of battery energy on transportation in the shop floor.

Although much related research has been completed, the problem of scheduling a single mobile robot with time windows, restricted capacity and multiple tasks have to be carried out during a planning horizon has received surprisingly little attention in the literature despite its important application in practice, e.g. part-feeding tasks. In this problem, a number of tasks with time windows have to be satisfied by the mobile robot. However, due to the limitation on carrying capacity, after satisfying some tasks, the robot has to return to a warehouse/base to load parts so that it can serve other tasks in the next route and so on (a route is from the warehouse, to locations of tasks, and back to the warehouse). In other words, the robot has to travel on a number of routes to perform tasks. The surveyed approaches are not well suited and cannot be directly used to solve this problem due to the lack of a mechanism handling both time-window and capacity constraints in case of the single robot. In the other previous work, Dang et al. (2012a) present a heuristic based on GA to solve the problem of sequencing feeding tasks. However, the heuristic is developed with a simple mechanism to adjust chromosomes, and the result of a randomly generated and simple numerical example is presented without rigorous comparisons with other methods. Therefore, in this paper an MIP model is first formulated to find optimal solutions for the problem. The MIP model is improved with some significant modifications from that in Dang et al. (2011). However, due to NP-hard nature of the problem the MIP model could only be applicable to small-scale problems in practice as its computation time significantly increases as the problem size grows. Hence, in order to deal with real-world applications, a heuristic based on GA, a promising algorithm to this class of problems with its powerful implicit parallelism and global search ability, is then extensively developed with an advanced constraint handling operator to effectively handle chromosomes so as to find near-optimal solutions. The quality of these solutions are then compared and evaluated by using the MIP solutions as reference points in a real-world case study and computational experiments.

Problem description

Automation technology in combination with advances in production management has dramatically changed the equipment used by manufacturing companies as well as the challenges in planning and control. With these changes, highly automated and unmanned production system have become more popular in several industries such as chemical and plastic, automotive, pump manufacturing, etc. A typical automatic production system includes intelligent and flexible machines and material handling device(s) grouped into cells in such a way that the entire production of a product can be carried out within one cell (Crama et al. 2000). In general a flexible manufacturing system with multiple machines, multiple storages and autonomous robots are taken into account. These autonomous robots are capable of transporting and feeding materials, tending machine, pre-assembling or inspecting quality. Thus, they have been widely employed in not only small companies, which focus on exact applications and a smaller range of products, but also large companies, which can diversify their applications in a longer term and larger range. For instances in an automobile parts manufacturing factory, a robot moves along a fixed track to load parts into pallets from a storage line. Likewise in an automated

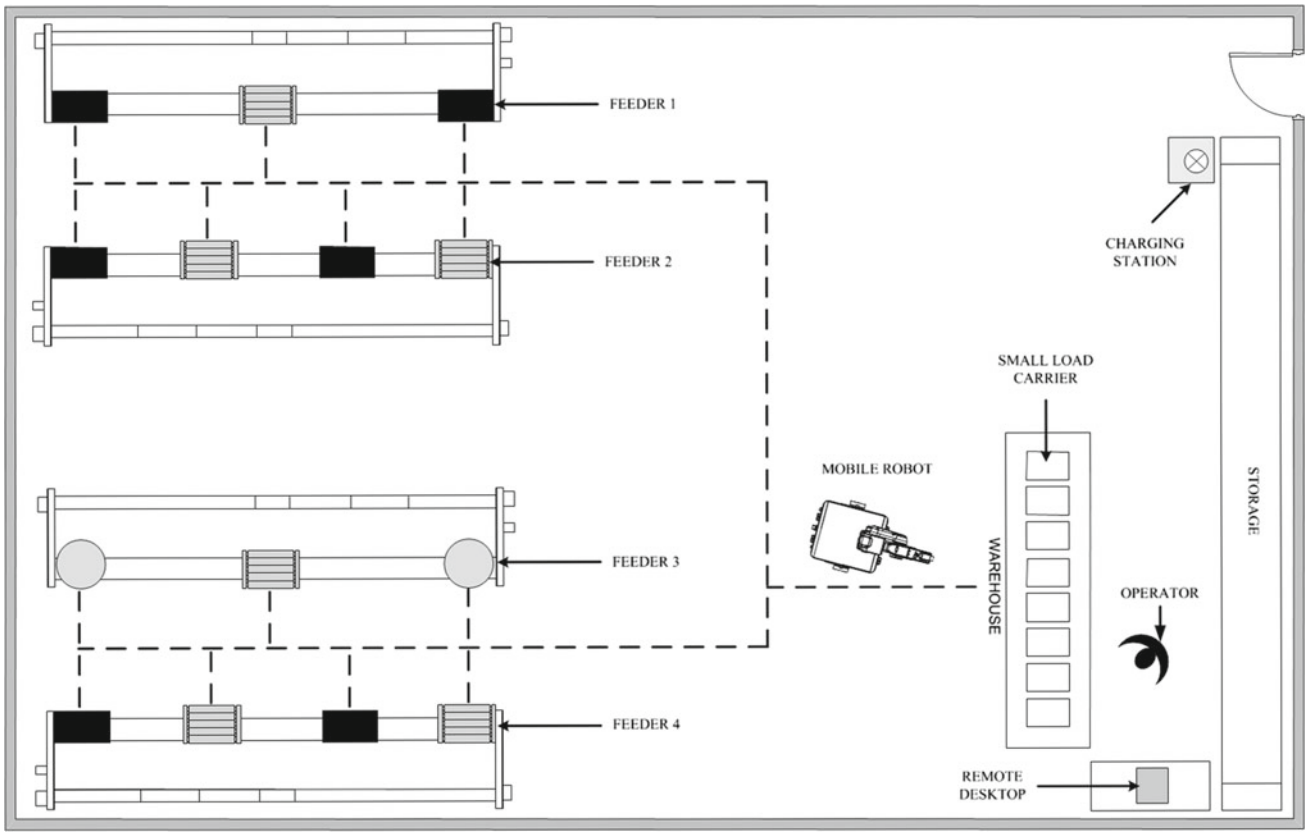

Fig. 1 Layout of the manufacturing cell

printed circuit board assembling factory, a robot has to transport the boards in-between insertion machines and buffers at the right time before the adhesive become dry (Maimon et al. 2000). Within the scope of this study, the interaction between a mobile robot and machines on production lines within a cell(s) is considered. Figure 1 below shows a typical layout of the manufacturing cell. The work is developed for a real cell that produces parts for the pump manufacturing industry at a factory.

Before assigning the mobile robot to the production environment, the manufacturing cell has one or several production lines which consist(s) of multiple machines. Feeders are designed to automatically supply parts to these machines. Pallets or boxes, which contain the parts, are placed next to these feeders. Part-feeding, the process of loading many parts at a time into feeders from the pallets or boxes, is a manually performed, non-value adding manufacturing task, and quite often disruptive (in-between and periodic) for production workers. Furthermore, when workers forget to fill the feeders, this may lead to stopping the production lines. A strategy that can reduce the dependence on human intervention for the part-feeding tasks is using an autonomous mobile robot instead of humans. However, to utilize the robot in this scenario requires changing the work environment and carefully scheduling part-feeding tasks of the robot (Dang et al. 2012b).

To meet the stated requirement the bartender concept is implemented. This also serves to make the part-feeding tasks performed by the robot more flexible and complete. In this concept every feeder is assigned the following characteristics: maximum level, minimum level and part-feeding rate to machine. Furthermore, instead of scattered pallets or boxes containing parts next to the feeders, a central warehouse (a bar) is created to gather different parts into one area. An operator (the bartender) put parts into small load carriers (SLCs) which are placed in the warehouse. The number of parts inside each SLC is equal to the difference between maximum level and minimum level of parts of the feeder in which that SLC is emptied. During operations, the robot will retrieve and carry one or several SLCs containing parts from the warehouse, move to feeder locations, empty all parts inside SLCs, return to the warehouse to unload all empty SLCs and take new filled SLCs. Each feeder has to be served a number of times in order to keep the production line in operation. The robot thus has a set of subtasks possessing time windows to carry out for each feeder during the planning horizon. In order to accomplish all the movements with the smallest consumed amount of battery energy and thereby increase the

availability of the robot, the total traveling time of the robot must be taken into account. Note that making decisions on which sequences the robot should supply parts to the feeders is part of the real-time operations of production planners. It means that the best solution must be quickly obtained at the beginning of production shifts or during the shifts due to errors in a manufacturing cell (e.g. machine breakdown) or changes in a manufacturing cell's conditions (e.g. cycle time of production lines). Moreover, as the problem is NP-hard, computation time exponentially grows with the size of the problem (e.g. larger number of feeders, longer planning horizon). It is therefore necessary to develop a computationally effective algorithm, namely a GA-based heuristic, to sequence the tasks in order to minimize its total traveling time while satisfying a number of technological constraints. To evaluate the performance of the proposed heuristic, an MIP model is formulated in the next section.

Mathematical formulation

In this section, a mixed-integer programming (MIP) model is developed to determine an optimal sequence in which the mobile robot visits n feeders to process part-feeding tasks. All feeding tasks, corresponding to deliveries of SLCs, are known in advance. A method based on the (s, Q) inventory system (Silver et al. 1998), to protect against shortage of parts over a replenishment lead time, is also presented to determine the time windows of the part-feeding tasks. Hard time windows scenario (Toth and Vigo 2002) will be taken into account in this model. The robot is based at a central warehouse and a limitation on the carrying capacity of the robot is imposed. The MIP model contains a number of decision variables that are constrained to have only integer values. Integer variables make optimization problems non-convex and thus far more difficult to solve. Memory and solution time may rise exponentially as the size of problem increases with more added integer variables. Therefore, in practice the MIP model could be applicable only to small-scale problems with few feeders on production line(s) and a short planning horizon—i.e. few tasks to schedule. For these scenarios, the MIP model will give optimal solutions which could be used as reference points to quantify the scale of benefits achieved by the GA-based heuristic (further developed in Genetic algorithm-based heuristic section). Assumptions, notations, time windows and the formulation of the MIP model are given in the following subsections.

Assumptions

- An autonomous mobile robot with a manipulation arm is taken into account in a disturbance free environment.
- The robot can carry one or several SLC(s) at a time.
- All tasks are periodic, independent and assigned to the same robot.
- Working time and travelling time of the robot between any pair of locations, in which either one of the locations could be a feeder or the warehouse, are known.
- The part-feeding rate to the machine of a feeder is known and constant.
- All feeders of machines have to be fed up to maximum levels and the robot starts from the warehouse at the initial stage.

Notations

N: set of all tasks ($N = \{0, 1, 2, \ldots, n\}$ where 0: task at the warehouse)
n_i: number of times task i has to be executed
R: set of all possible routes ($R = \{1, 2, \ldots, R_{max}\}$, $R_{max} = \Sigma n_i, \forall i \in N \backslash \{0\}$)
e_{ik}: release time of subtask k of task i
d_{ik}: due time of subtask k of task i
p_i: periodic time of task i
w_i: working time of robot at location of task i
t_{ij}: travelling time of the robot from location of task i to location of task j
c_i: part-feeding rate to machine of feeder i
v_i: minimum level of parts in feeder i
u_i: maximum level of parts in feeder i
Q_m: maximum number of SLCs could be carried by robot
T: planning horizon

Decision variables:

$$x_{ik}^{jlr} = \begin{cases} 1 & \text{if robot travels from location of task } i \text{ with subtask } k \text{ to location of task } j \text{ with subtask } l \text{ in route } r \\ 0 & \text{otherwise} \end{cases}$$

y_{ik}: route index to which subtask k of task i belongs
s_{ik}: starting time of subtask k of task i

Time windows

A (s, Q) inventory policy is used with the defined characteristics of the feeders to determine the hard time windows of the part-feeding tasks as shown in Fig. 2, Eqs. (1), (2), and (3) below.

$$p_i = (u_i - v_i)\, c_i, \quad \forall i \in N \backslash \{0\} \tag{1}$$

$$e_{ik} = e_{ik-1} + p_i, \ \forall i \in N \backslash \{0\}, \ k \in \{1, 2, \ldots, n_i\}, \ e_{i0} = 0 \tag{2}$$

$$d_{ik} = e_{ik} + (v_i - 0)\, c_i, \ \forall i \in N \backslash \{0\}, \ k \in \{1, 2, \ldots, n_i\} \tag{3}$$

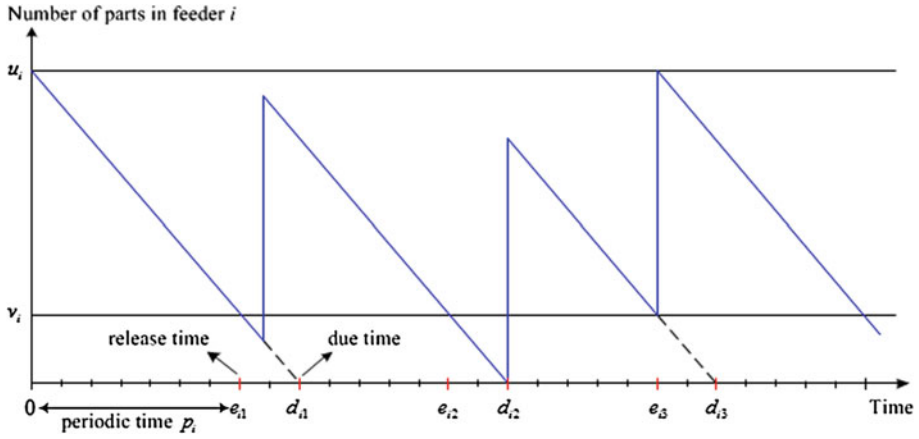

Fig. 2 Time windows of part-feeding tasks based on the (s, Q) inventory system

Because of the periodic characteristics of the tasks for feeder i whose periodic time is calculated as in Eq. (1) a number of subtasks must be carried out. The number of subtasks of task i is defined as: $n_i = \lfloor T/p_i \rfloor$. The robot must start processing a subtask k of task i within the associated hard time window of that subtask. It means that the robot is not allowed to arrive at feeder i after the upper bound of the time window. If the robot arrives at feeder i before the lower bound of the time window it will wait to begin service. The lower bound of the time window, or release time of subtask k of task i, is set to the time when the number of parts inside feeder i drops to a certain level v_i [Eq. (2)]. The upper bound of the time window, or due time of subtask k of task i, is defined to the time when there are no parts in feeder i [Eq. (3)].

Mixed-integer programming model

$$\text{Objective function: min} \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} t_{ij}x_{ik}^{jlr} \tag{4}$$

Subject to:

$$e_{ik} \le s_{ik} \le d_{ik} \quad \forall i \in N\backslash\{0\},\ k \in \{1, 2, \ldots, n_i\} \tag{5}$$

$$\sum_{j\in N\backslash\{0\}}\sum_{l=1}^{n_j} x_{01}^{jl1} = 1 \tag{6}$$

$$\sum_{j\in N\backslash\{0\}}\sum_{l=1}^{n_j}\sum_{r\in R} x_{01}^{jlr} \le 1 \tag{7}$$

$$\sum_{(i,k),(j,l)\in Z} x_{ik}^{jlr} \le |Z| - 1 \quad \forall r \in R, \quad \forall Z \subseteq Z_T,$$
$$Z_T = \{(i,k) | i \in N\backslash\{0\},\ k \in \{1, 2, \ldots, n_i\}\} \tag{8}$$

$$\sum_{j\in N}\sum_{l=1}^{n_j}\sum_{r\in R} x_{ik}^{jlr} = 1 \quad \forall i \in N\backslash\{0\},\ k \in \{1, 2, \ldots, n_i\} \tag{9}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} x_{ik}^{jlr} = 1 \quad \forall j \in N\backslash\{0\},\ l \in \{1, 2, \ldots, n_j\} \tag{10}$$

$$\sum_{i\in N}\sum_{k=1}^{n_i}\sum_{j\in N\backslash\{0\}}\sum_{l=1}^{n_j} x_{ik}^{jlr} \le Q_m \quad \forall r \in R \tag{11}$$

$$s_{ik} + \left(w_i + t_{ij}\sum_{r\in R} x_{ik}^{jlr}\right) - L\left(1 - \sum_{r\in R} x_{ik}^{jlr}\right) + (y_{jl} - y_{ik}) \times (t_{i0} + w_0 + t_{0j} - t_{ij}) \le s_{jl}$$
$$\forall i, j \in N,\quad k \in \{1, 2, \ldots, n_i\},\ l \in \{1, 2, \ldots, n_j\} \tag{12}$$

$$y_{jl} = \sum_{i\in N}\sum_{k=1}^{n_i}\sum_{r\in R} r \times x_{ik}^{jlr} \quad \forall j \in N\backslash\{0\},\ l \in \{1, 2, \ldots, n_j\} \tag{13}$$

$$y_{jl} \ge y_{ik}\sum_{r\in R} x_{ik}^{jlr} \quad \forall i, j \in N,\ k \in \{1, 2, \ldots, n_i\},\ l \in \{1, 2, \ldots, n_j\} \tag{14}$$

$$x_{ik}^{jlr} \in \{0, 1\} \quad \forall r \in R,\ \forall i, j \in N,\ k \in \{1, 2, \ldots, n_i\},\ l \in \{1, 2, \ldots, n_j\} \tag{15}$$

$$y_{ik}\text{: positive integer variable} \quad \forall i \in N\backslash\{0\},\ k \in \{1, 2, \ldots, n_i\} \tag{16}$$

The objective function (4) minimizes the total traveling time of the robot. Constraint (5) ensures that the starting time of any subtask of a task satisfies the time window of that subtask. Constraints (6) and (7) ensure that the robot starts from the warehouse at the initial stage. Constraint (8) eliminates the sub-tours among subtasks of tasks, where Z is a subset of Z_T, where Z_T is a set of all subtasks of tasks at feeders and the warehouse. Constraints (9) and (10) ensure that a subtask of a task is completed exactly once. Constraint (11) forbids the robot to load a higher number of SLCs than its maximum capacity in the number of SLC Q_m. Constraint (12) handles the traveling time requirements between any pair of subtasks of tasks, where L is a given sufficiently large

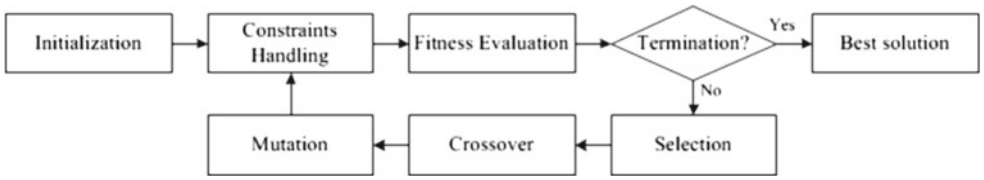

Fig. 3 Flowchart of genetic algorithm-based heuristic

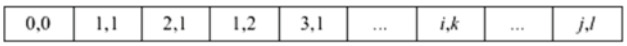

Fig. 4 Genetic representation

constant. In case two subtasks of the same task or different tasks are connected but are not in the same route, the robot should visit the warehouse to unload empty SLCs and load filled ones. Constraint (13) assigns a subtask of a task to a route and constraint (14) guarantees an ascending sequence of route indices for subtasks of tasks. Constraints (15) and (16) imply the types of variables.

Genetic algorithm-based heuristic

In this section a genetic algorithm, a search method using the principle of biological evolution (Goldberg 1989), is used to develop a heuristic. The developed heuristic allows converting the presented problem so that near-optimal solutions can be found. The genetic algorithm-based heuristic shown in Fig. 3 consists of the following main steps: genetic representation and initialization; constraints handling and fitness evaluation; genetic operators including selection, crossover and mutation; termination criteria.

Genetic presentation and initialization

For the problem under consideration, the natural path representation (Potvin 1996) is used to represent a chromosome or a solution, which represents an ordering of subtasks of tasks of the robot as shown in Fig. 4 below. Each gene in the chromosome consists of two factors. The first factor refers to a task while the second factor implies a subtask of that task. The original length of the chromosome is equal to the total number of subtasks of part-feeding tasks added to the first subtask of warehouse task ($1 + \Sigma n_i$).

For the initial generation, the first factors of genes on a chromosome are randomly filled with tasks at feeders. The frequency of a task is the number of times which that task has to be executed. The second factors of genes having the same first factor/same task are filled in ascending order of subtasks of that task.

Constraints handling and fitness evaluation

After initialization or crossover and mutation operations, chromosomes are adjusted to be valid and their fitness values are calculated. A valid chromosome should satisfy the two constraints: time windows of subtasks of tasks and limitation on the carrying capacity Q_m of the robot. For the first constraints, starting time of a subtask of a task should be in-between release time and due time of that subtask. The second constraint requires the robot does not serve more subtasks than the number of SLCs it is carrying. An approach of handling these constraints is developed and applied to each chromosome in the initial or descendant generations as shown in Fig. 5 along with the description below.

Step 1: Rearrange genes possessing the same first factor (task) in ascending order of their second factor (subtask). Note that, Step 1 is not employed in the initial generation due to the outcome of the initialization procedure.

Step 2: Temporarily rearrange genes considering only time windows.

Step 2.1: Compute starting time and check the satisfaction of time window constraints for gene (j, l) at position p preceded by gene (i, k) at position $p - 1$.
$s_{jl} = s_{ik} + w_i + t_{ij} \times 1_{\{i \neq j\}}$
If $s_{jl} \leq d_{jl}$ then $s_{jl} = s_{ik} + w_i + t_{ij} \times 1_{\{i \neq j\}}$ or $s_{jl} = e_{jl} \times 1_{\{sjl \leq ejl\}}$, and check if p is the last position on the chromosome. If not, then go to Step 2.6. If so, then go to Step 3.
Otherwise, go to Step 2.2.

Step 2.2: Considering genes from position 1 to the position p on the chromosome, make a set A of positions of genes whose due times are greater than that of gene (j, l).

Step 2.3: Insert gene (j, l) to a position p_a randomly selected from set A.

Step 2.4: Re-compute starting times and check the satisfaction of the time windows constraints for genes from the position p_a to the position p.
If starting time of any gene from the position p_a to the position p does not satisfy its time window,

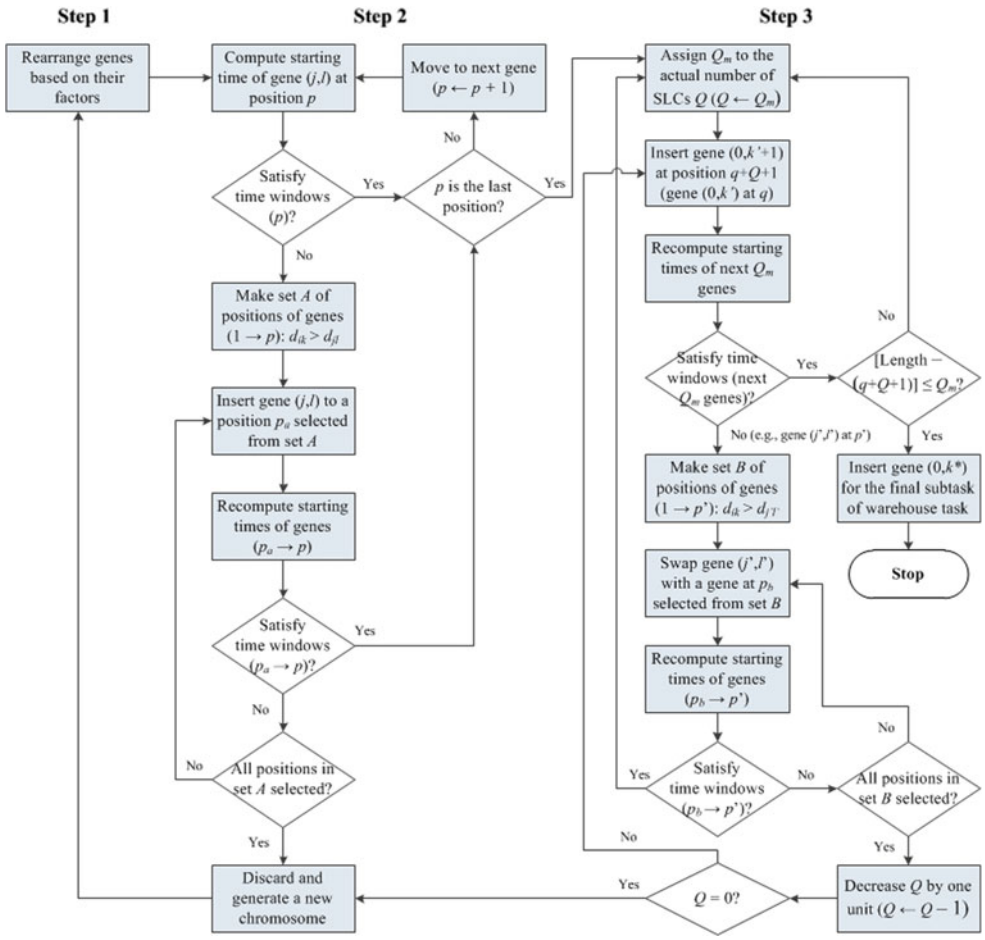

Fig. 5 Flowchart of handling constraints

then check if all positions in set A are selected. If not, then go to Step 2.3. If so, then go to Step 2.5.

Otherwise, check if p is the last position on the chromosome. If not, then go to Step 2.6. If so, then go to Step 3.

Step 2.5: Discard the chromosome, generate a new one instead and go back to Step 1.

Step 2.6: Move to the next gene at position $p+1$ on the chromosome ($p \leftarrow p+1$) and go back to Step 2.1.

Step 3: Rearrange genes considering limitation on carrying capacity of the robot and time windows.

Step 3.1: Assign the maximum number of SLCs Q_m to the actual number of SLCs, denoted as Q, carried by the robot in a route ($Q \leftarrow Q_m$).

Step 3.2: Considering gene $(0, k')$, which is a subtask of a warehouse task, at a position q on the chromosome, insert gene $(0, k'+1)$ at position $q+Q+1$. Note that, the chromosome length is increased by one unit after each insertion.

Step 3.3: Re-compute starting times and check the satisfaction of the time windows constraints for next Q_m genes.

If starting time of any gene among the next Q_m genes does not satisfy its time window, then go to Step 3.4.

Otherwise, check if [length of chromosome—$(q+Q+1)$] $\leq Q_m$. If not, then go back to Step 3.1. If so, then go to Step 3.8.

Step 3.4: Considering genes from position 1 to position p' of a gene (j', l') which does not satisfy its time window among the next Q_m genes, make a set B of positions of genes whose due times are greater than that of gene (j', l').

Step 3.5: Swap gene (j', l') with a gene at a position p_b randomly selected from set B.

Step 3.6: Re-compute starting times and check the satisfaction of time windows constraints for genes from the position p_b to the position p'.

If the starting time of any gene from the position p_b to the position p' does not satisfy its time window, then check if all positions in set B are

selected. If not, then go back to Step 3.5. If so, then go to Step 3.7.
Otherwise, go to Step 3.1.

Step 3.7: Decrease Q by one unit ($Q \leftarrow Q-1$), and check if Q is 0. If not, then go back to Step 3.2. If so, go back to Step 2.5.

Step 3.8: Insert gene $(0, k^*)$ representing the final subtask of a warehouse task at the end of the chromosome.

Following the approach of handling constraints, the fitness evaluation will take place. The fitness value of a chromosome is equal to the total travelling time of the robot, Σt_{ij}, where i, j are the first factors of genes on the chromosome.

Genetic operators

Selection, crossover and mutation are three main genetic operators. For selection, various evolutionary methods can be applied to this problem. $(\mu + \lambda)$ selection is used to choose chromosomes for reproduction. Under this method, μ parents and λ offspring compete for survival and the μ best out of the set of offspring and old parents, i.e. the μ lowest in term of the total travelling time, are selected as the parents of the next generation. This selection mechanism guarantees that the best solutions up to now are always in the parent generation (Dang et al. 2012a).

Crossover operator generates offspring by combining the information contained in the parent chromosomes so that the offspring will have the desirable features from their parents. The Roulette-wheel selection is used in the algorithm, which probabilistically selects the parent chromosomes based on their fitness values (Ho and Ji 2004; Moon et al. 2006). Owing to the nature of the considered our minimization problem, the higher the total travelling time of the robot, the less fitness a chromosome should show. Let $TR(p)$ denote the total travelling time of the robot under the solution represented by parent p. The fitness value of parent p is defined as: $F_p = \max\{TR(p) : 1 \le p \le N_p\} - TR(p)$ where N_p: population size. The expected probability of parent p to be selected is $F_p / \Sigma F_p$.

There are many different crossover operators that can be used on chromosomes. These are represented by the path presentation, e.g. partially-mapped crossover (PMX), cycle crossover (CX), order crossover (OX), order-based crossover (OBX), and position-based crossover (PBX) (Tsujimura and Gen 1999; Lin et al. 2006). Although the crossover operators may affect the efficiency of the search process, the quality of solutions is often reasonably close. In the presented experiment, OX will be used to generate an offspring as described below. Genes which represent subtasks of the warehouse task are removed before two cut points are randomly chosen on the parent chromosomes. The string between these cut points in one of the parents is first copied to the offspring. The remaining positions are then filled by considering the sequence of genes in the other parent starting after the second cut point. When reaching to the end of the offspring, the sequence continues at position 1. The OX operates with probability P_c. Figure 6 illustrates the OX procedure.

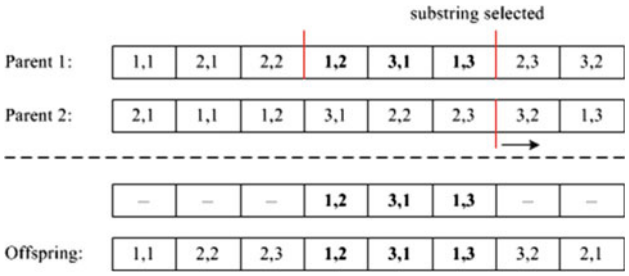

Fig. 6 Example of order crossover procedure

Whenever an offspring is produced, it undergoes a mutation operator which is applied with probability P_m. The mutation selects two genes within the offspring at random and then swaps these genes to produce heterogeneous chromosomes. This procedure avoids premature convergence of the GA-based heuristic. Note that offspring produced after crossover and mutation operations might not be valid. Therefore they have to be adjusted by using the approach of handling constraints described above.

Termination criteria

Termination criteria are employed to determine when the GA-based heuristic should be stopped. On the one hand, computation time plays an important role in practice because taking decisions on which sequences the robot should serve feeders is a part of the real-time operations of production planners. In other words, the best solution is required to be quickly obtained. Therefore, if the best solutions over generations do not converge, the maximum computation time CT_m would be used to stop the run. On the other hand, if the best solution does not improve over G_c consecutive generations, it would not be valuable to continue searching. The best solution up to now is then returned as the near-optimal solution. However, it should be noted that high-quality local optima might exist (in case of existing feasible solutions) because of the combinatorial nature of the problem.

Numerical examples

To examine the performance of the MIP model and GA-based heuristic, a case study and computational experiments are conducted in this section. However, before carrying out these cases, the statistical analysis to evaluate and set parameter values of the GA-based heuristic is first presented. Then,

the case study including two different demonstrations has been investigated with real data of Grundfos A/S, a Danish company which is one of the world's leading pump manufacturers. A part of Grundfos production facilities, CR (compression ratio) 1-2-3 impeller line at CR factory, has been used to implement these demonstrations. The extension of the case study considering several reasonable assumptions has been also conducted to make the evaluation of both approaches more convincing. Finally, various problem instances are randomly generated and tested in order to provide more persuasive evidence of the performance of the proposed heuristic. The MIP model has been coded and solved by the mathematical modelling language ILOG CPLEX, while the proposed heuristic has been programmed in VB.NET. All the experiment run on a PC having an Intel®Core i5 2.67 GHz processor and 4 GB RAM.

Statistical analysis of GA parameters

The GA parameters have effects on the efficiency of the GA-based heuristic. The statistical analysis is hence carried out with a randomly generated problem instance to examine the effects and set the values for these parameters. The statistical analysis concerned N_p, P_c, P_m, G_c, and CT_m (μ and λ are not included in this analysis because μ and λ are dependent on N_p and P_c, in other words, μ is the population size N_p while λ is the average number of offspring $P_c \times N_p$). Each parameter is tested at four different levels. These are respectively: N_p (50, 100, 150, 200), P_c (0.4, 0.6, 0.8, 1.0), P_m (0.05, 0.1, 0.15, 0.2), G_c (50, 100, 150, 200), and CT_m in seconds (15, 30, 45, 60). There are ten observations under each level. The 40 runs of each parameter are made in random order to prevent the effects of unknown nuisance variables. In order to determine if a parameter has effect on the objective value of the proposed heuristic, the analysis of variance (ANOVA) is performed to test the null hypothesis H_0 which is that there are no differences in means of objective values at all four levels of the parameter. In addition, to set a value for a parameter, the averages and standard deviations of the objective value and computation time at four levels of the parameter, which have been obtained from the test of 40 runs, are considered. By comparing the averages to each other under consideration of the standard deviations, the most appropriate value of the parameter can be set. For instance, Table 1 shows the ANOVA for parameter N_p while Table 2 shows the averages and standard deviations of the objective value and computation time at the four levels of N_p.

As seen from Table 1, since $F = 8.207 > 2.866$, we reject H_0 and conclude that means of objective values differ; that is, the population size N_p significantly affects the objective value of the proposed heuristic. It also can be seen from Table 2 that there is a considerable improvement in the average of the objective value when N_p increases from 50 to 100. However, only slight improvements in the objective value are achieved with much more computation time when N_p increases from 100 to 150 and 200. Furthermore, the standard deviation of the objective value at N_p of 100 is better than the other values of N_p. The value of N_p is hence set to be 100. By performing the statistical analyses in the same manner for the other parameters, it also proved that P_c, P_m, G_c, and CT_m affect the objective values of the proposed heuristic, and the values of P_c, P_m, G_c, and CT_m are set to be 0.8, 0.1, 100, and 60, respectively (for details see Appendix). These parameter values are used in the "Case study" and "Computational experiments" sections.

Table 1 Analysis of variance (ANOVA) for population size N_p

Source of variation	*SS*	*df*	*MS*	*F*	*P* value	$F_{0.05,3,36}$
N_p	26,099	3	8,700	8.207	0.0003	2.866
Error	38,161	36	1,060			
Total	64,260	39				

Table 2 Objective value and computation time at four levels of population size N_p

N_p	Objective value (s)		Computation time	
	Average	SD	Average	SD
50	2,108	35	0.928	0.185
100	2,061	29	2.161	0.497
150	2,049	32	3.110	0.753
200	2,044	34	4.907	1.075

Case study

The chosen area for the case study is the CR 1-2-3 impeller production line shown in Fig. 8 that manufactures impellers for the CR pumps. The CR line consists of a warehouse and four feeders that have to be served by the mobile robot. The warehouse is indexed 0 and the feeders are indexed from 1 to 4 ($N = \{0, 1, 2, 3, 4\}$) and named Back Plate, Van Feeder 1, Van Feeder 2, and Front Plate, respectively. Furthermore, different feeders are filled by different types of parts, namely back plates for feeder 1, vanes for feeder 2 and 3 and front plates for feeder 4. To produce an impeller on the CR 1-2-3 line these three types of parts, are automatically assembled. The impeller consists of six vanes with three from feeder 2 and the other three from feeder 3, one front plate from feeder 4 and one back plate from feeder 1. Figure 7 shows the different parts of an impeller, while Fig. 8 particularly illustrates the production area where the case study has been implemented. The safety fences and warning signs are used as depicted in Fig. 8 to ensure that no people enter the area

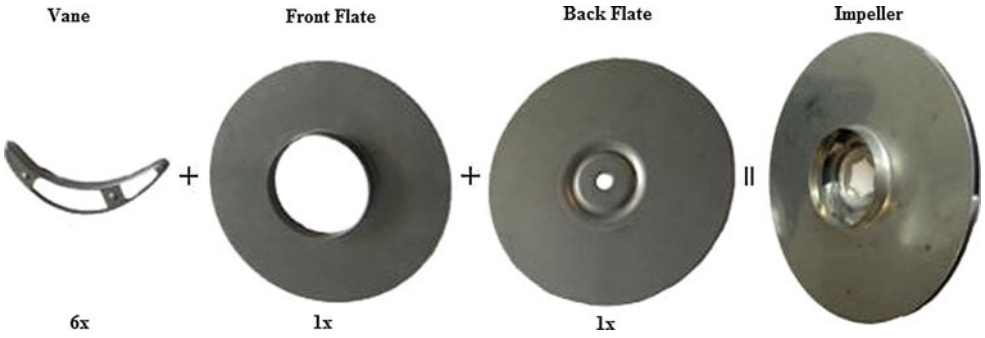

Fig. 7 Different parts of an impeller produced on the CR 1-2-3 line

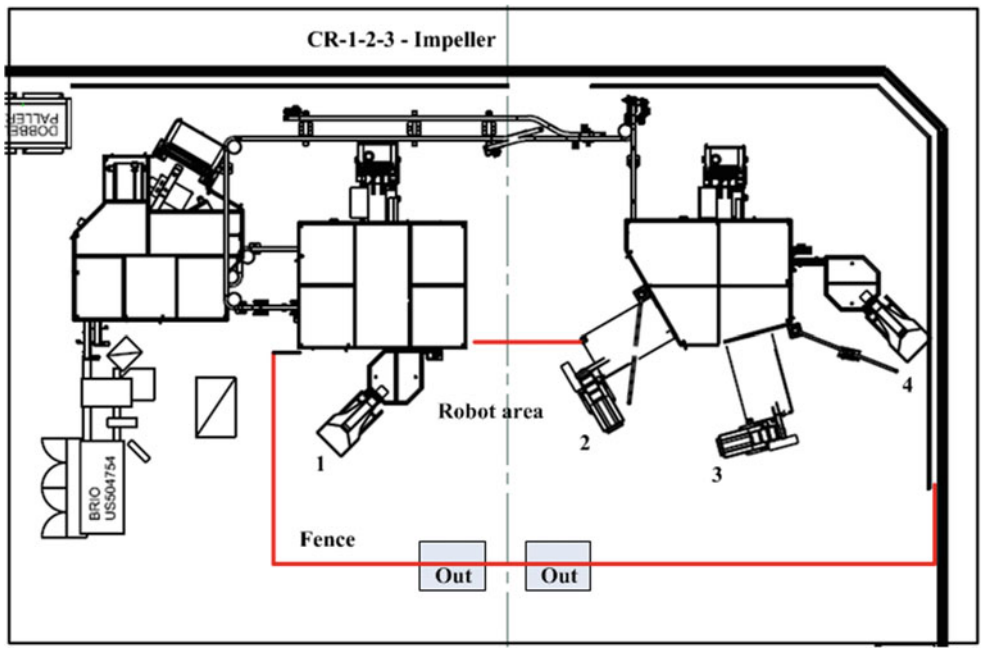

Fig. 8 CR 1-2-3 impeller production line

as well as to prevent the mobile robot leaving that area while the demonstrations are taking place.

The following data are taken from the Manufacturing Execution System (MES) as well as real tests on the shop floor of the CR factory. This data is used as input for two demonstrations of the case study. The average number of parts per SLC fed to feeder 1 or 4 is 125 (approximately 2 kg/SLC), while the average number of parts per SLC fed to feeder 2 and 3 is 1,100 (approximately 1 kg/SLC). The maximum levels, minimum levels, and part-feeding rates to machines of feeders are given in Table 3. The part-feeding rates to machines of feeders are derived from the cycle time of the CR 1-2-3 line of 4.5 s (Reuther et al. 2010). Specifically feeders 1 and 4 feed machines with one back plate and one front plate every 4.5 s, while feeders 2 and 3 feed machines with one vane every 1.5 s (3 vanes for every 4.5 s). The working times of the robot at the feeders are given in Tables 4, and 5 shows the travelling times of the robot from locations to locations (feeder 0 means the warehouse).

Table 3 Maximum levels, minimum levels, and part-feeding rates to machines of feeders

Feeder/task	1	2	3	4
Maximum level (part)	250	2,000	2,000	250
Minimum level (part)	125	900	900	125
Part-feeding rate (second/part)	4.5	1.5	1.5	4.5

Table 4 Working times of robot at locations (seconds)

Feeder/task	0	1	2	3	4
Working time of robot	90	42	42	42	42

In the initial design, the mobile robot has the capability to carry up to three SLCs at a time while performing part-feeding tasks at the feeders. Hence, two different demonstrations of the case study have been investigated corresponding to two maximum numbers of SLCs, $Q_m = 2$ and $Q_m = 3$, and with the planning horizon T of approximately 45 min due to the robot's battery limitation. The parameters p_i, e_{ik}, and d_{ik} are respectively calculated based on Eq. (1), (2), and (3) with the data in Table 3. As a result, the total number of subtask of part-feeding tasks is 10 and the number of decision variables is 4,040. In each demonstration of the case study, two cases of the MIP are investigated for comparing the performances with the GA-based heuristic. The first case is carried out when giving the same maximum computation time CT_m (60 s) as the GA-based heuristic while the second case is performed without limitation on the computation/run time. Table 6 gives solutions of the MIP and proposed heuristic on the objective value and computation time (in seconds) for the demonstrations at the CR factory. The cells containing a "–" symbol indicate that the results of the corresponding problems cannot be obtained by using the corresponding approach. Sequences of part-feeding tasks found by the heuristic are depicted using Gantt charts in Fig. 9.

From Table 6, it can be seen that when giving the same maximum computation time CT_m as the GA-based heuristic, the MIP is not able to find any feasible solution. In contrast

Table 5 Traveling times of robot from locations to others (seconds)

From feeder	To feeder				
	0	1	2	3	4
0	0	34	37	34	40
1	39	0	17	34	50
2	35	17	0	35	49
3	34	33	35	0	47
4	36	47	48	46	0

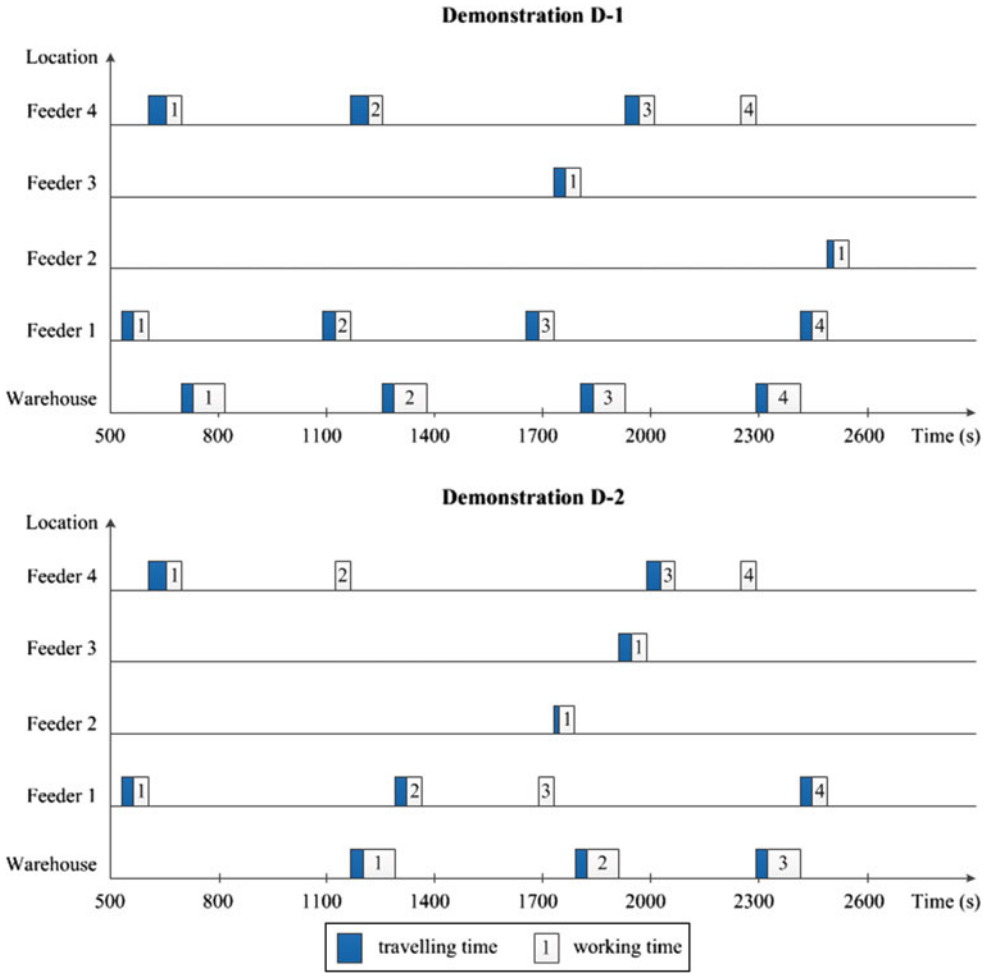

Fig. 9 Gantt charts for the solutions of two demonstrations of the case study

Table 6 Solutions of the case study under MIP and GA-based heuristic

Demo	Q_m	MIP (limited to CT_m)		MIP (unlimited)		GA-based heuristic		Penalty of the heuristic (%) versus MIP (unlimited)
		Objective value (s)	Computation time (s)	Objective value (s)	Computation time (s)	Objective value (s)	Computation time (s)	
D-1	2	–	–	488	21,589.34	504	<1	3.28
D-2	3	–	–	384	8,377.27	396	<1	3.13

Table 7 Comparison between MIP and GA-based heuristic for extension of the case study

Q_m	T (hour)	Total subtasks of tasks	Number of variables	MIP		GA-based heuristic			
				Objective value (s)	Computation time (s)	Objective value (s)		Computation time (s)	
						Average	SD	Average	SD
2	1	16	16, 448	–	–	827	0	0.35	0.04
	2	32	131, 200	–	–	1, 596	2	1.16	0.10
	4	66	1, 150, 248	–	–	3, 205	39	5.01	1.37
	8	136	10, 062, 368	–	–	6, 738	70	25.09	7.28
3	1	16	16, 448	–	–	648	7	0.34	0.06
	2	32	131, 200	–	–	1, 231	17	1.32	0.27
	4	66	1, 150, 248	–	–	2, 615	44	4.96	0.99
	8	136	10, 062, 368	–	–	5, 667	83	16.46	3.98
4	1	16	16, 448	–	–	628	4	0.31	0.06
	2	32	131, 200	–	–	1, 206	6	1.05	0.15
	4	66	1, 150, 248	–	–	2, 497	30	4.24	1.21
	8	136	10, 062, 368	–	–	5, 223	85	15.49	2.36

the proposed heuristic found solutions for the demonstrations (objective values of 504 s/8.4 min in D-1 and 396 s/6.6 min in D-2). These objective values found through the heuristic are greater than those found by the MIP when the run time of the MIP is unlimited. However, the differences are only about 3 % and this is deemed to be an acceptable error. Furthermore, the computation time shows that use of the MIP is too time-consuming while the proposed heuristic significantly faster obtains near-optimal solutions (approximately 6 h in D-1 or 2.3 h in D-2 as opposed to less than a second). It also reveals that the higher maximum numbers of SLCs the robot can carry, the less it has to travel around the manufacturing cell (8.4 min with Q_m of 2 as opposed to 6.6 min with Q_m of 3).

To make the evaluation more convincing, the case study is extended by assuming that the robot has the capability of carrying up to 4 SLCs at a time and the battery limitation of the robot allows it to work up to 8 h (a full production shift). Further comparisons of the objective value and computation time for the MIP and GA-based heuristic are presented in Table 7. Note that in this extended experiment, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs. The total number subtask of all tasks and number of decision variables are also given. From Table 7, it can be seen that the proposed heuristic has the capability of solving larger problems while the MIP cannot find any feasible solution for problems of this scale. It also shows that in case of full production shift of 8 h, the proposed heuristic is able to find the best solutions in <30 s. Moreover, the standard deviation of the objective value is quite small in comparison with the average. The GA-based heuristic, therefore, demonstrates efficiency in solving the larger problems.

Computational experiments

In this section, the performance of the proposed heuristic will be tested on a large number of problem instances. 20 problems are generated with different numbers of feeders, maximum numbers of SLCs, planning horizons and other system parameters. The number of feeders and the maximum number of SLCs are randomly generated in the ranges of [3, 20]

Table 8 Comparison between MIP and GA-based heuristic for 20 randomly generated problems

No.	Number of feeders	Q_m	T (hour)	Total subtasks of tasks	Number of variables	MIP		GA-based heuristic			
						Objective value (s)	Computation time (s)	Objective value (s)		Computation time (s)	
								Average	SD	Average	SD
1	7	3	1	10	4, 040	790	60.00	377	9	0.22	0.04
2	15	2	1	25	62, 600	–	–	986	14	1.52	0.19
3	6	2	1	15	13, 560	–	–	775	10	0.29	0.03
4	3	4	1	6	888	184	4.93	184	0	0.14	0.08
5	16	3	1	36	186, 768	–	–	1, 120	22	2.61	0.39
6	5	2	2	28	87, 920	–	–	1, 287	7	0.76	0.15
7	11	3	2	53	595, 720	–	–	1, 657	33	4.00	0.84
8	10	4	2	44	340, 912	–	–	1, 006	31	2.78	0.50
9	4	3	2	13	8, 840	811	60.00	428	0	0.18	0.01
10	18	4	2	70	1, 372, 280	–	–	2, 156	47	39.72	5.34
11	15	3	4	106	4, 764, 488	–	–	3, 504	76	28.83	6.07
12	12	4	4	99	3, 881, 592	–	–	2, 998	59	14.61	1.98
13	9	4	4	84	2, 371, 152	–	–	2, 229	38	5.97	0.91
14	19	3	4	115	6, 083, 960	–	–	4, 732	33	60.00	0.00
15	11	2	4	92	3, 115, 120	–	–	4, 070	28	30.73	3.58
16	8	3	8	113	5, 772, 040	–	–	4, 498	79	13.18	2.29
17	13	2	8	154	14, 609, 672	–	–	6, 696	71	42.89	7.47
18	17	4	8	200	32, 000, 800	–	–	7, 342	95	60.00	0.00
19	14	3	8	170	19, 652, 680	–	–	6, 249	83	48.75	5.08
20	20	4	8	220	42, 592, 880	–	–	8, 071	99	60.00	0.00

and [2, 4], respectively. The planning horizons in hours are 1, 2, 4, and 8 (corresponding to an eighth, a quarter, half, and full of the production shift). The maximum and minimum levels of parts in feeders are respectively uniformly distributed within the ranges of [500, 2,000] and [100, 1,000] while part-feeding rates to machines of feeders (in seconds) are generated in the interval [1.5, 6.5]. The working times of the robot in seconds at feeders and the warehouse are respectively distributed within the range of [40, 60] and [80, 100] while the traveling times of the robot in seconds are generated in the interval [20, 60]. Note that the time/cost matrices of the generated traveling times should satisfy the triangle inequality. The comparisons between the MIP and GA-based heuristic for 20 randomly generated problems are presented in Table 8. Similar to the extension of the case study, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs. General information for these 20 problems is also shown in Table 8.

It can be observed from Table 8 that the GA-based heuristic is superior to the MIP for large problems. The MIP found feasible solutions for problem instances 1, 4, and 9. However, the solutions found by the MIP are much worse than those found by the proposed heuristic (except problem instance 4 in which both approaches found the optimal solution). In addition, the MIP cannot find any feasible solution for the other problems. The GA-based heuristic, by contrast, is able to find best solutions for all 20 problem instances. As the size of the problem increases especially the total subtasks of all tasks, the computation time of the proposed heuristic becomes longer. The results also show that with the same planning horizon, the same or nearly the same number of feeders, the less maximum number of SLCs the robot can carry, the more difficult it is for the proposed heuristic to generate feasible sequences of part-feeding tasks, thus the longer computation time it may require to obtain best solutions (e.g. problem instances 12 and 15). Furthermore, in terms of the objective value, the standard deviation is quite small in comparison with the average. These results provide

more persuasive evidence to prove that the GA-based heuristic performs effectively.

Conclusion

This paper presents the results of a study of the novel problem of scheduling a single mobile robot in order to perform part-feeding tasks on production lines. The robot has been assigned to the production by applying the bartender concept. This permits the part-feeding tasks to be properly performed by the robot. To accomplish all tasks in a planning horizon within the allowable limit of battery capacity and power, it is important for production planners to determine feeding sequences which minimize the total travelling time of the robot. This must be done while taking into account a number of practical constraints. The main novelty of this research lies in the simultaneous consideration of hard time windows of tasks and limitation of carrying capacity on the single mobile robot. A mixed-integer programming model to find exact optimal solutions for the problem was developed. Due to the NP-hard nature of the problem this solution approach is only applicable to small-scale problems with few feeders and a short planning horizon. A genetic algorithm-based heuristic was then proposed to find near-optimal solutions. The quality of these solutions could then be evaluated by using the MIP solutions as reference points to quantify the scale of benefits. The real case study at an impeller production line and more computational experiments were described to demonstrate the effectiveness of both approaches. The results of the case study showed that use of the MIP was too time-consuming while the proposed heuristic was significantly faster in obtaining near-optimal solutions. Further experiments provided persuasive evidence that the proposed heuristic is capable of solving problems of various sizes and more efficient than the MIP in terms of the objective value when giving the same maximum computation time. It can be also observed that the larger number of SLCs the robot can carry, the easier the proposed heuristic can generate feasible sequences and the less computation time may be required to obtain the best solution. These solutions are useful to managers for decision making at operational levels and the proposed heuristic could be also be applied in a variety of tasks of not only mobile robots but also automatic guided vehicles or unmanned aerial vehicles. For further research, a general model of scheduling multiple mobile robots should be considered. This will also include a rescheduling mechanisms based on the obtained schedules and feedback from the mobile robot fleet and shop floor. This will enable to deal with real-time disturbances such as machine breakdown or unexpected shortage of parts of feeders.

Acknowledgments This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

Appendix

- Probability of crossover P_c (Tables 9, 10).

The average and standard deviation of the objective value is the best when P_c is equal to 0.8 or 1.0. However, only slight improvement on the objective value is achieved with longer computation time when P_c increases from 0.8 to 1.0. The value of P_c is hence set to be 0.8.

- Probability of mutation P_m (Tables 11, 12).

There are improvements on the average of the objective value when P_m increases from 0.05 to 0.2. However, these

Table 9 ANOVA for P_c

Source of variation	*SS*	*df*	*MS*	*F*	*P* value	$F_{0.05,\ 3,36}$
P_c	15,158	3	5,053	4.673	0.0074	2.866
Error	38,922	36	1,081			
Total	54,080	39				

Table 10 Objective value and computation time at four levels of P_c

P_c	Objective value (s)		Computation time	
	Average	SD	Average	SD
0.4	2,099	35	0.980	0.258
0.6	2,077	36	1.232	0.218
0.8	2,053	29	2.103	0.528
1.0	2,051	30	2.470	0.611

Table 11 ANOVA for P_m

Source of variation	*SS*	*df*	*MS*	*F*	*P* value	$F_{0.05,3,36}$
P_m	17,827	3	5,942	5.235	0.0042	2.866
Error	40,862	36	1,135			
Total	58,689	39				

Table 12 Objective value and computation time at four levels of P_m

P_m	Objective value (s)		Computation time	
	Average	SD	Average	SD
0.05	2,107	36	1.034	0.353
0.10	2,088	31	2.100	0.294
0.15	2,073	39	2.691	0.467
0.20	2,049	29	3.904	0.807

Table 13 ANOVA for G_c

Source of variation	*SS*	*df*	*MS*	*F*	*P* value	$F_{0.05,3,36}$
G_c	10,609	3	3,536	4.924	0.0057	2.866
Error	25,855	36	718			
Total	36,464	39				

Table 14 Objective value and computation time at four levels of G_c

G_c	Objective value (s)		Computation time	
	Average	SD	Average	SD
50	2,090	31	0.865	0.188
100	2,058	24	1.393	0.312
150	2,055	24	1.668	0.261
200	2,047	27	2.136	0.256

Table 15 ANOVA for CT_m

Source of variation	*SS*	*df*	*MS*	*F*	*P* value	$F_{0.05,3,36}$
CT_m	548,131	3	182,710	22.560	2×10^{-8}	2.866
Error	291,556	36	8,099			
Total	839,687	39				

Table 16 Objective value and computation time at four levels of CT_m

CT_m	Objective value (s)		Computation time	
	Average	SD	Average	SD
15	6,382	119	15	0
30	6,185	69	30	0
45	6,118	87	45	0
60	6,077	77	60	0

improvements are minor in comparison with the increase of the computation time. Furthermore, the standard deviations of the objective value at P_m of 0.1 and 0.2 are better than the other values of P_m. The value of P_m is hence set to be 0.1.

- No-improvement consecutive generations G_c (Tables 13, 14).

  Similar to the case of N_p, the value of G_c is set to be 100.

- Maximum computation time CT_m (Tables 15, 16).

There are considerable improvements on the average of the objective value when CT_m increases from 15 to 60 s (the best average is achieved at CT_m of 60). Furthermore, the standard deviation of the objective value at CT_m of 60 is better than the other values of CT_m except the case of 30. Note that the value of CT_m is set based on the improvements on the objective value because the ANOVA of CT_m shows that this parameter strongly affects the objective value ($F = 22.560 > 2.866$). The value of CT_m is hence set to be 60 (a problem instance of larger size is randomly generated to test CT_m).

References

Ascheuer, N., Escudero, L. F., Grotschel, M., & Stoer, M. A. (1993). Cutting plane approach to the sequential ordering problem (with application to job scheduling in manufacturing). *SIAM Journal of Optimization*, *3*(1), 25–42.

Askin, R. G., & Stanridge, C. R. (1993). *Modeling and analysis of manufacturing systems*. New York: Wiley.

Carlton, W. B., & Barnes, J. W. (1996). Solving the traveling salesman problem with time windows using tabu search. *IIE Transactions*, *28*(8), 617–629.

Carpaneto, G., & Toth, P. (1980). Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Management Science*, *26*(7), 736–743.

Carpaneto, G., Dell'Amico, M., & Toth, P. (1995). Exact solution of large-scale, asymmetric traveling salesman problems. *ACM Transactions on Mathematical Software*, *21*(4), 394–409.

Chatterjee, S., Carrera, C., & Lynch, L. A. (1996). Genetic algorithms and traveling salesman problems. *European Journal of Operational Research*, *9*, 490–510.

Chen, S. M., & Chien, C. Y. (2011). Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. *Expert Systems with Applications*, *38*(12), 14439–14450.

Chen, C., & Tseng, C. (1996). The path and location planning of workpieces by genetic algorithms. *Journal of Intelligent Manufacturing*, *7*(1), 69–76.

Choi, I. C., Kim, S. I., & Kim, H. S. (2003). A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem. *Computers and Operations Research*, *30*(5), 773–786.

Crama, Y., Kats, V., van de Klundert, J., & Levner, E. (2000). Cyclic scheduling in robotic flow shops. *Annals of Operations Research*, *96*(1–4), 97–124.

Dang, Q. V., Nielsen, I., & Steger-Jensen, K. (2011). Scheduling a single mobile robot for feeding tasks in a manufacturing cell. In *Proceedings of international conference advances in production management systems*, Stavanger, Norway

Dang, Q. V., Nielsen, I., & Bocewicz, G. (2012a). Trends in PAAMS, AISC 157. In J. M. C. Rodríguez (Ed.), *A genetic algorithm-based heuristic for part-feeding mobile robot scheduling problem* (pp. 85–92). Berlin, Heidelberg: Springer.

Dang, Q. V., Nielsen, I., & Steger-Jensen, K. (2012b). Ecoproduction & logistics—new trends and business practices. In P. Golinska (Ed.), *Scheduling a single mobile robot incorporated into production environment* (pp. 185–201). Berlin, Heidelberg: Springer.

Edan, Y., Flash, T., Peiper, U. M., Shmullevich, I., & Sarig, Y. (1991). Near-minimum-time task planning for fruit-picking robots. *IEEE Transactions on Robotics and Automation*, *7*(1), 48–55.

Geng, X., Chen, Z., Yang, W., Shi, D., & Zhao, K. (2011). Solving the travelling salesman problem based on an adaptive simulated annealing algorithm with greedy search. *Applied Soft Computing*, *11*(4), 3680–3689.

Germs, R., Goldengorin, B., & Turkensteen, M. (2012). Lower tolerance-based branch and bound algorithms for the ATSP. *Computer and Operations Research*, *39*(2), 291–298.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. New York: Addison-Wesley.

Han, M. H., McGinnis, L. F., Shieh, J. S., & White, J. A. (1987). On sequencing retrievals in an automated storage/retrieval system. *IIE Transactions*, *19*(1), 56–66.

Hasegawa, M., & Ikeguchi, T. (2002). Solving large scale traveling salesman problems by chaotic neurodynamics. *Neural Networks*, *15*(2), 271–283.

Ho, W., & Ji, P. (2004). A hybrid genetic algorithm for component sequencing and feeder arrangement. *Journal of Intelligent Manufacturing*, *15*(3), 307–315.

Hurink, J., & Knust, S. (2002). A tabu search algorithm for scheduling a single robot in a job-shop environment. *Discrete Applied Mathematics*, *119*(1–2), 181–203.

Landrieu, A., Mati, Y., & Binder, Z. (2001). A tabu search heuristic for the single vehicle pickup and delivery problem with time windows. *Journal of Intelligent Manufacturing*, *12*(5–6), 497–508.

Lin, L., Shinn, S. W., Gen, M., & Hwang, H. (2006). Network model and effective evolutionary approach for AGV dispatching in manufacturing system. *Journal of Intelligent Manufacturing*, *17*(4), 465–477.

Liu, F., & Zeng, G. (2009). Study of genetic algorithm with reinforcement learning to solve the TSP. *Expert Systems with Applications*, *36*(3), 6995–7001.

López-Ibáñez, M., & Blum, C. (2010). Beam-ACO for the travelling salesman problem with time windows. *Computers and Operations Research*, *37*(9), 1570–1583.

Maimon, O., Braha, D., & Seth, V. (2000). A neural network approach for a robot task sequencing problem. *Artificial Intelligence in Engineering*, *14*(2), 175–189.

Miller, D. L., & Pekny, J. F. (1991). Exact solution of large asymmetric traveling salesman problems. *Science*, *251*(4995), 754–761.

Moon, C., Kim, J., Choi, G., & Seo, Y. (2002). An efficient genetic algorithm for the traveling salesman problem with precedence constraints. *European Journal of Operational Research*, *140*(3), 606–617.

Moon, C., Seo, Y., Yun, Y., & Gen, M. (2006). Adaptive genetic algorithm for advanced planning in manufacturing supply chain. *Journal of Intelligent Manufacturing*, *17*(4), 509–522.

Ohlmann, J. W., & Thomas, B. W. (2007). A compressed-annealing heuristic for the traveling salesman problem with time windows. *INFORMS Journal on Computing*, *19*(1), 80–90.

Potvin, J. Y. (1996). Genetic algorithms for the traveling salesman problem. *Annals of Operation Research*, *63*(3), 339–370.

Ratliff, H. D., & Rosenthal, A. S. (1983). Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research*, *31*(3), 507–521.

Reuther, H., Simon, B. & Mads, H. (2010). 3D simulation used for evaluating the use and implementation of the mobile manipulator "Little Helper" at Grundfos A/S. In H. Hvolby (Ed.), *Proceedings of the 12th international modern information technology in the innovation processes of the industrial enterprises (MITIP) conference, 29–31 August 2010* (pp. 9–18). Aalborg: Centre for Logistics, Aalborg University.

Silver, E. A., Pyke, D. F., & Peterson, R. (1998). *Inventory management and production planning and scheduling*. New York: Wiley.

Snyder, L. V., & Daskin, M. S. (2006). A random-key genetic algorithm for the generalized traveling salesman problem. *European Journal of Operational Research*, *174*(1), 38–53.

Suárez, R., & Rosell, J. (2005). Feeding sequence selection in a manufacturing cell with four parallel machines. *Robotics and Computer-Integrated Manufacturing*, *21*(3), 185–195.

Syslo, M. M., Deo, N., & Kowalik, J. S. (1983). *Discrete optimization algorithms: With Pascal programs*. New Jersey: Prentice-Hall.

Toth, P., & Vigo, D. (2002). *The vehicle routing problem*. Philadelphia: SIAM.

Tsai, C. F., Tsai, C. W., & Tseng, C. C. (2004). A new hybrid heuristic approach for solving large traveling salesman problem. *Information Sciences*, *166*(1–4), 67–81.

Tsujimura, Y., & Gen, M. (1999). Parts loading scheduling in a flexible forging machine using an advanced genetic algorithm. *Journal of Intelligent Manufacturing*, *10*(2), 149–159.

Turkensteen, M., Ghosh, D., Goldengorin, B., & Sierksma, G. (2008). Tolerance-based branch and bound algorithms for the ATSP. *European Journal of Operational Research*, *189*(3), 775–788.

Xing, L. N., Chen, Y. W., Yang, K. W., Hou, F., Shen, X. S., & Cai, H. P. (2008). A hybrid approach combining an improved genetic algorithm and optimization strategies for the asymmetric traveling salesman problem. *Engineering Applications of Artificial Intelligence*, *21*(8), 1370–1380.

Zacharia, P Th, & Aspragathos, N. A. (2005). Optimal robot task scheduling based on genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, *21*(1), 67–79.

Paper G

Multi-objective genetic algorithm for real-world mobile robot scheduling problem

This paper has been published in APMS 2012, Part I, IFIP AICT 397.
Springer-Verlag Berlin Heidelberg, pp. 518–525

Multi-objective Genetic Algorithm for Real-World Mobile Robot Scheduling Problem

Quang-Vinh Dang, Izabela Nielsen, and Kenn Steger-Jensen

Department of Mechanical and Manufacturing Engineering, Aalborg University,
Fibigerstræde 16, 9220 Aalborg, Denmark
{vinhise,izabela,kenn}@m-tech.aau.dk

Abstract. This paper deals with the problem of scheduling feeding tasks of a single mobile robot which has capability of supplying parts to feeders on production lines. The performance criterion is to minimize the total traveling time of the robot and the total tardiness of the feeding tasks being scheduled, simultaneously. In operation, the feeders have to be replenished a number of times so as to maintain the manufacture of products during a planning horizon. A method based on predefined characteristics of the feeders is presented to generate dynamic time windows of the feeding tasks which are dependent on starting times of previous replenishment. A heuristic based on genetic algorithm which could be used to produce schedules in online production mode is proposed to quickly obtain efficient solutions. Several numerical examples are conducted to demonstrate results of the proposed approach.

Keywords: Multi-objective, Scheduling, Mobile Robot, Genetic Algorithm.

1 Introduction

The automation technology in combination with advances in production management has dramatically changed the equipment used by manufacturing companies as well as the issues in planning and control. With these changes, highly automated and unmanned production systems have become more popular in several industrial areas, e.g., automotive, robot, and pump manufacturing [3]. An automatic production system consists of intelligent and flexible machines and mobile robots grouped into cells in such a way that entire production of each product can be performed within one of the cells. With embedded batteries and manipulation arms, mobile robots are capable of performing various tasks such as transporting and feeding materials, tending machines, pre-assembling, or inspecting quality at different workstations. They have been thus employed in not only small companies which focus on exact applications and a small range of products, but also large companies which can diversify applications in a longer term and larger range. Within the scope of this study, a given problem is particularly considered for a single mobile robot which will automate part-feeding tasks by not only transporting but also collecting containers of parts and emptying them into the feeders needed. However, to utilize mobile robots in an efficient manner requires the ability to properly schedule feeding tasks. Hence, it is

C. Emmanouilidis, M. Taisch, D. Kiritsis (Eds.): APMS 2012, Part I, IFIP AICT 397, pp. 518–525, 2013.

important to plan in which sequence mobile robots process feeding operations so that they could effectively work while satisfying a number of practical constraints.

The problem of scheduling part-feeding tasks of the mobile robot has been modeled in some respects comparable to the Asymmetric Traveling Salesman Problem (ATSP) which belongs to the class of NP-hard combinatorial optimization problems [7]. Among heuristic approaches, Genetic Algorithm (GA) has been widely used in the research areas of TSP, ATSP, or robot task-sequencing problems. Liu and Zheng [10], Moon et al. [12], and Snyder and Daskin [13] discussed about using GAs to solve TSP, while Choi et al. [2] and Xing et al. [14] proposed GAs to deal with ATSP. Zacharia and Aspragathos [15] introduced a method based on GA and an innovative encoding to determine the optimal sequence of manipulator's task points which is considered an extension to the TSP. Beside genetic algorithms, Bocewiz [1] presented the knowledge-based and constraint programming-driven methodology in planning and scheduling of multi-robot in a multi-product job shop taking into account imprecise activity specifications and resource sharing. Hurink and Knust [9] proposed a tabu search algorithm for scheduling a single robot in a job-shop environment considering time windows and additionally generalized precedence constraints. Maimon et al. [11] also presented a neural network approach with successful implementation for the robot task-sequencing problem.

Although there are many related research, the problem of scheduling a single mobile robot with dynamic time windows and restricted capacity where multiple routes have to be carried out has surprisingly received little attention in the literature despite its important applications in practice, e.g. part-feeding task. Such a task must be executed a number of times within time windows which are dependent on starting times of the previous executions of that task, hence, the term, dynamic time windows. The objectives of minimizing the total traveling time of the robot and the total tardiness of the tasks are taken into account to support the global objective of maximizing system throughput. The existing approaches are not well suited and cannot be directly used to solve the problem. Thus, in this paper, a heuristic based on GA, a possibly promising approach to the class of multi-objective optimization, is developed to find efficient solutions for the problem. The advantageous feature of GA is the multiple directional and global search by maintaining a population of potential solutions from generation to generation. Such population-to-population approach is useful to explore all non-dominated solutions of the problem [6].

The remainder of this paper is organized as follows: in the next section, problem statement is described while a genetic algorithm-based heuristic is presented in Section 3. Numerical examples are conducted to demonstrate results of the proposed approach in Section 4. Finally, conclusions are drawn in Section 5.

2 Problem Statement

The work is developed for a cell which produces parts or components for the pump manufacturing industry at a factory in Denmark. The essential elements considered in the manufacturing cell consist of an autonomous mobile robot with limitation on carrying capacity, a central warehouse designed to store small load carriers (SLCs), and multiple feeders designed to automatically feed parts to machines of production

lines. Besides, every feeder has three main characteristics including maximum level, minimum level, and part-feeding rate to machine. In operation, the robot will retrieve and carry one or several SLCs containing parts from the warehouse, move to feeder locations, empty all parts inside SLCs, then return to the warehouse to unload empty SLCs and load filled ones. To maintain the manufacture of a quantity of products during a given planning horizon, the feeders (tasks) have to be replenished a number of times, the robot consequently has a set of subtasks of tasks to be carried out within time windows. Such a time window of a subtask of a task could be only determined after starting time of the previous subtask of that task. Fig. 1 below shows a layout of the described manufacturing cell.

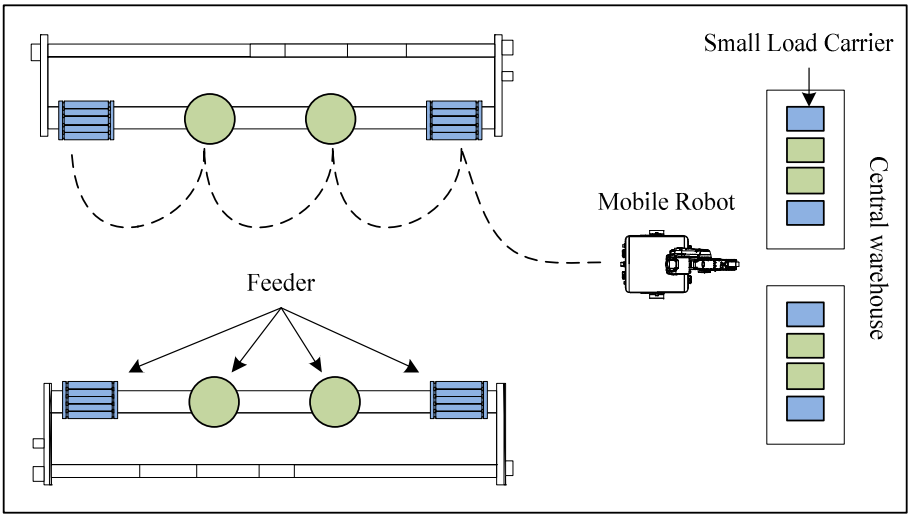

Fig. 1. Layout of the manufacturing cell

To enable the construction of a feeding schedule for the mobile robot, assumptions are considered as follows:

- The robot can carry one or several SLC(s) at a time.
- All tasks are periodic, independent, and assigned to the same robot.
- Working time, traveling time between any pairs of locations of the robot, and part-feeding rate to machine of a feeder are known.
- All feeders of machines must be fed up to maximum levels and the robot starts from the ware house at the initial stage.

In order to accomplish all the movements with a smallest consumed mount of battery energy, the total traveling time of the robot is an important objective to be considered. Apart from that, another performance measure is the amount of time a feeder has been waiting to be replenished by the robot. Alternatively, due time of a time window of a feeding task could be considered soft constraint, i.e. schedules that do not meet this constraint are taken into account. In addition, making decisions on which way the robot should provide parts to feeders is a part of real-time operations of production

planners. Moreover, concerning the problem belong to NP-hard class, computation time exponentially grows with the size of the problem (e.g. larger number of feeders). It is therefore necessary to develop a computationally effective algorithm, namely GA-based heuristic, which determines in which sequence the feeders should be supplied so as to minimize the total traveling time of the robot and the total tardiness of feeding tasks while satisfying a number of practical constraints.

3 Genetic Algorithm-Based Heuristic

In this section, genetic algorithm, a random search method taking over the principle of biological evolution [8], is applied to develop a heuristic which is allowed to convert the aforementioned problem to the way that efficient solutions could be found. The GA-based heuristic shown in Fig. 2 comprises of the following components: genetic representation and initialization; constraint handling and fitness assignment; genetic operators including selection, crossover, and mutation; termination criteria.

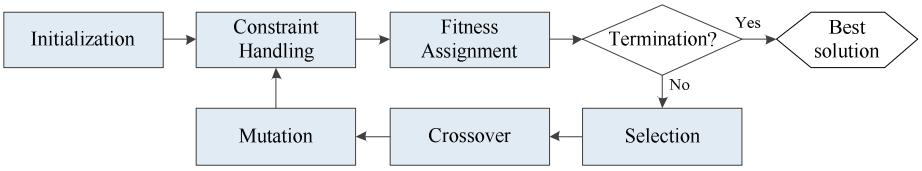

Fig. 2. Flow chart of GA-based heuristic

3.1 Genetic Representation and Initialization

For the problem under consideration, a solution can be represented by a chromosome of non-negative integers $(0, i, j, \dots, i, \dots, k)$ which is an ordering of part-feeding tasks of the robot where i, j, k: feeder index; $i, j, k = 1 \div n$; n: number of feeders. The original length of a chromosome is equal to the total number of subtasks of tasks added the first subtask of task at the central warehouse $(1 + \sum n_i\,; n_i$: number of subtasks of task $i)$.

For the initial generation, genes on a chromosome are randomly filled with tasks at feeders. The frequency of such a task is the number of subtasks of that task, in other words, number of times that tasks has to be executed.

3.2 Constraint Handling and Fitness Assignment

After initialization or crossover and mutation operations, chromosomes are handled to be valid and then assigned fitness values. A valid chromosome should satisfy two constraints of limitation on carrying capacity Q_m of the robot and time windows of subtasks of part-feeding tasks. For the first type of constraints, to guarantee the robot not to serve more number of feeders than number of SLCs carried in one route, the subtasks of task at the warehouse represented by zeroes are inserted into a chromosome after every Q_m genes starting from the first gene. For instances, if the

limitation on carrying capacity of the robot is two SLCs of parts, the chromosome should be restructured to be $(0, i, j, 0, \ldots, 0, k, l, 0, \ldots, 0)$.

The second type of constraints requires a subtask of a task to be started after release time and completed by the due time of that subtask, if possible. As mentioned, due time constraints are considered soft constraints. They thus could be modeled as an objective of the total tardiness of part-feeding tasks. The release time and due time could be determined as shown in Equation (1) and (2) below.

$$r_{ik+1} = s_{ik} + (u_i - v_i) \times c_i, i = 1 \div n, k = 1 \div n_i \quad (1)$$

$$d_{ik} = r_{ik} + (v_i \times c_i), i = 1 \div n, k = 1 \div n_i \quad (2)$$

where r_{ik}, d_{ik}, s_{ik}: release time, due time, and starting time of subtask k of task i
u_i, v_i, c_i: maximum level, minimum level of parts of feeder i, and part-feeding rate to machine of feeder i

After constraint handling procedure, the objectives of the total traveling time of the robot and the total tardiness of part-feeding tasks are calculated one after another for every chromosome in the population. A weighted-sum fitness function F is then used to assign a fitness value to each chromosome as shown in Equation (3) where t_{ij} is traveling time of the robot from one location to another, w_i is working time of the robot per SLC at feeder i, and α is the weighted coefficient.

$$F = \alpha \times \left(\sum_{i,j} t_{ij}\right) + (1 - \alpha) \times \left(\sum_{i,k} (\max\{0, (s_{ik} + w_i/2) - d_{ik}\})\right) \quad (3)$$

3.3 Genetic Operators

Selection, crossover, and mutation are three main genetic operators. For selection, various evolutionary methods could be applied in this problem. $(\mu + \lambda)$ selection is used to choose chromosomes for reproduction. Such selection mechanism guarantees that the best solutions up to now are always in the parent generation [4-5].

Crossover operator generates offspring by combining the information contained the parent chromosomes so that the offspring inherits good features from their parents. The Roulette-wheel selection is used to select the parent chromosomes based on their weighted-sum fitness values. Order crossover (OX) [6] operated with probability P_c will be employed to generate an offspring as follows. Genes having zero values are removed before two cut points are randomly chosen on the parent chromosomes. A string between these cut points in one of the parent chromosomes is first copied to the offspring, the remaining positions are then filled according to the sequence of genes in the other parent starting after the second cut point. When an offspring is produced, it undergoes insertion mutation [6] with probability P_m which selects a gene at random and inserts it in a random position.

3.4 Termination Criteria

Termination criteria are employed to determine when the GA-based heuristic should be stopped. Note that making decisions on which sequences the robot should serve feeders is a part of real-time operations of production planners. Therefore, on the one

hand if the best solutions over generations do not converge to a value, the maximum generation G_m would be used to stop the run. On the other hand, if the best solution does not improve over G_c consecutive generations, it would not be valuable to continue searching.

4 Numerical Examples

The performance of the GA-based heuristic will be tested on several problem instances in this section. Three problems, which are as similar to the real-world case as they can be, are generated with difference number of feeders (namely 3, 5, and 10 feeders), and other system parameters such as limitation on carrying capacity, working time, traveling time of the mobile robot, planning horizon, and characteristics of feeders. The robot is designed to carry up to 3 SLCs at a time to perform part-feeding tasks during a given planning horizon of one hour (corresponding to an eighth of a full production shift). The maximum and minimum levels of parts of feeders are respectively distributed within the ranges of [300, 2000] and [100, 1000] while part-feeding rates in seconds are in-between the interval [1.5, 4.5]. The working times of the robots in seconds at feeders and the warehouse per SLC are respectively distributed within the range of [40, 60] and [25, 40] while the traveling times of the robot in seconds are in-between the interval [20, 60]. Note that the cost matrix of the generated traveling times should satisfy the triangle inequality.

For GA parameters, the population size, P_c, P_m, G_m, and G_c are set to be 100, 0.6, 0.2, 500, and 100, respectively. The weighted-sum fitness function F (Equation 3) will be calculated using one of three different values of the weight coefficient α, namely, 0.2, 0.5, and 0.8. The proposed heuristic has been coded in VB.NET, and all the problem instances run on a PC having an Intel® Core i5 2.67 GHz processor and 4 GB RAM. The results for three randomly generated problems in combination with three values of the weighted coefficient α are presented in Table 1 below.

Table 1. The best solutions of three generated problems

Problem	No. of feeder	No. of subtasks of tasks	Weighted coefficient (α)	Total traveling time of robot (second)	Total tardiness of tasks (second)	Computation time (second)
1	3	11	0.2	432	0	0,49
			0.5	432	0	0,63
			0.8	428	6	0,41
2	5	24	0.2	706	0	1,00
			0.5	690	4	1,12
			0.8	682	15	1,10
3	10	42	0.2	1564	0	2,52
			0.5	1542	22	2,70
			0.8	1528	55	2,40

The total traveling time of the robot, total tardiness of tasks, and computation time shown in Table 1 are the average of 10 runs. It can be observed that as the weighted coefficient α increases, two objectives of each problem instance have opposite trends where the total traveling time of the robot decreases and the total tardiness of tasks increases. In other words, as saving battery energy allowing the robot to be utilized in a longer duration is more important, the robot has a tendency to travel less and vice versa. Similar explanation is also applicable to the total tardiness of part-feeding tasks. Such kinds of solutions in Table 1 are non-dominated solutions for which no improvement in any objective function is possible without sacrificing the other objective function. It also shows that when the size of the problems grows, the computation time of the GA-based heuristic becomes longer, but it is still acceptable (i.e., the largest problem size with 10 feeders and the coefficient α of 0.5 requires 2.7 seconds in average to find the efficient solution). These results provide evidence to prove that the GA-based heuristics could be used to produce efficient schedules within reasonable time in online production mode.

The above solutions are initial schedules for the robot. These schedules serve as input to a Mission Planner and Control (MPC) program which is accessed by using XML-based TCP/IP communication to interact with the robot, Manufacturing Execution System (MES), and the module of GA-based heuristic. In practice, there might be some errors in manufacturing such as machine breakdown, or changes in manufacturing conditions such as characteristics of feeders (e.g. minimum levels of parts), or carrying capacity of the robot. These events will be reported by the MES so that the MPC program can update current states of the shop floor and then call the heuristic module to reschedule part-feeding tasks of the robot. By relaxing the last assumption mentioned in Section 2, the proposed heuristic in turn will use the current states as new input, re-optimize to get alternative schedules, and send these schedules back to the MPC program.

5 Conclusions

In this paper, a problem of scheduling a single mobile robot to carry out part-feeding tasks of production lines is studied. To maintain the manufacture of products, it is important for planners to determine feeding sequences which minimize the total traveling time of the robot and the total tardiness of the feeding tasks while taking into account a number of practical constraints. The main novelty of this research lies in the consideration of dynamic time windows and limitation on carrying capacity where multiple routes have to be performed by the single mobile robot. A genetic algorithm-based heuristic was proposed to find efficient solutions for the problem. The results in the numerical examples showed that the proposed heuristic is fast enough to be used to generate efficient schedules compromising the objectives in online production mode. The heuristic may be also used to produce alternative schedules in rescheduling scenarios when there might be some errors or changes in manufacturing conditions. Moreover, the heuristic could be considered to deal with more performance criteria according to requirements of planners, and by investigating different scenarios with various weighted coefficients of those criteria, it can specify which schemes are more

beneficial for the manufacturing. For further research, a general model of scheduling multiple mobile robots should be considered together with rescheduling mechanisms to deal with real-time disturbances.

Acknowledgments. This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

References

1. Bocewicz, G., Bach, I., Wójcik, R.: Production Flow Prototyping subject to Imprecise Activity Specification. Kybernetes 38, 1298–1316 (2009)
2. Choi, I.C., Kim, S.I., Kim, H.S.: A Genetic Algorithm with a Mixed Region Search for the Asymmetric Traveling Salesman Problem. Comput. Oper. Res. 30, 773–786 (2003)
3. Crama, Y., Kats, V., van de Klundert, J., Levner, E.: Cyclic Scheduling in Robotic Flowshops. Ann. Oper. Res. 96, 97–124 (2000)
4. Dang, Q.-V., Nielsen, I.E., Bocewicz, G.: A Genetic Algorithm-Based Heuristic for Part-Feeding Mobile Robot Scheduling Problem. In: Rodríguez, J.M.C., Pérez, J.B., Golinska, P., Giroux, S., Corchuelo, R. (eds.) Trends in PAAMS. AISC, vol. 157, pp. 85–92. Springer, Heidelberg (2012)
5. Dang, Q.V., Nielsen, I., Steger-Jensen, K., Madsen, O.: Scheduling a Single Mobile Robot for Part-feeding Tasks of Production Lines. In: J. Intell. Manuf. (2012) (accepted)
6. Gen, M., Lin, L.: Network Models and Optimization. Springer, London (2008)
7. Germs, R., Goldengorin, B., Turkensteen, M.: Lower Tolerance-Based Branch and Bound Algorithms for the ATSP. Comput. Oper. Res. 39, 291–298 (2012)
8. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York (1989)
9. Hurink, J., Knust, S.: A TabuSearch Algorithm for Scheduling a Single Robot in a Job-shop Environment. Discrete Appl. Math. 119, 181–203 (2002)
10. Liu, F., Zeng, G.: Study of Genetic Algorithm with Reinforcement Learning to Solve the TSP. Expert Syst. Appl. 36, 6995–7001 (2009)
11. Maimon, O., Braha, D., Seth, V.: A Neural Network Approach for a Robot Task Sequencing Problem. Artif. Intell. Eng. 14, 175–189 (2000)
12. Moon, C., Kim, J., Choi, G., Seo, Y.: An Efficient Genetic Algorithm for the Traveling Salesman Problem with Precedence Constraints. Eur. J. Oper. Res. 140, 606–617 (2002)
13. Snyder, L.V., Daskin, M.S.: A Random-Key Genetic Algorithm for the Generalized Traveling Salesman Problem. Eur. J. Oper. Res. 174, 38–53 (2006)
14. Xing, L.N., Chen, Y.W., Yang, K.W., Hou, F., et al.: A Hybrid Approach Combining an Improved Genetic Algorithm and Optimization Strategies for the Asymmetric Traveling Salesman Problem. Eng. Appl. Artif. Intell. 21, 1370–1380 (2008)
15. Zacharia, P.T., Aspragathos, N.A.: Optimal Robot Task Scheduling Based on Genetic Algorithms. Robot Comput. Integrated Manuf. 21, 67–79 (2005)

Paper H

Modelling and scheduling autonomous mobile robot for a real-world industrial application

This paper has been published in Proceedings of IFAC Conference on Manufacturing Modelling, Management and Control, Saint Petersburg, Russia, Volume 7, Part 1, pp. 2098-2103

Preprints of the 2013 IFAC Conference on Manufacturing Modelling, Management, and Control, Saint Petersburg State University and Saint Petersburg National Research University of Information Technologies, Mechanics, and Optics, Saint Petersburg, Russia, June 19-21, 2013

Modelling and Scheduling Autonomous Mobile Robot for a Real-World Industrial Application

Quang-Vinh Dang*. Izabela Nielsen*
Simon Bøgh*. Grzegorz Bocewicz.**

**Department of Mechanical and Manufacturing Engineering, Aalborg University, Aalborg East, Denmark (e-mail: vinhise@m-tech.aau.dk, sb@m-tech.aau.dk, izabela@m-tech.aau.dk)*
***Department of Computer Science and Management, Koszalin University of Technology, Poland (e-mail: bocewicz@ie.tu.koszalin.pl)*

Abstract: The paper deals with a real-world implementation of an autonomous industrial mobile robot performing an industrial application at a pump manufacturing factory. In the implementation, the multi-criteria optimization problem of scheduling tasks of a mobile robot is taken into account. The paper proposes an approach composing of: a mobile robot system design ("Little Helper"), an appropriate and comprehensive industrial application (multiple-part feeding tasks), an implementation concept for industrial environments (the bartender concept), and a real-time heuristics integrated into Mission Planning and Control software to schedule the mobile robot in the industrial application. Results from the real-world implementation show that "Little Helper" is capable of successfully serving four part feeders in three production cells within a given planning horizon using the best schedule generated from the real-time heuristics. The results also demonstrated that the proposed real-time heuristics is capable of finding the best schedule in online production mode.

Keywords: Robotics, Modelling, Scheduling, Heuristics, Pumps.

1. INTRODUCTION

The shift in paradigm from mass production to customized production and the resumption of production in wage-intensive countries have created needs for flexibility, transformability and cost-efficiency, especially in the field of automation and robotics (Jovan et al., 2003). Robots are widely used in industry to perform 4D tasks; dumb, dangerous, dull, and/or dirty. Therefore, industrial robotics forms an essential part of the manufacturing backbone, but the robots of today are rather inflexible, as they are often dedicated and in fixed positions (High Level Group, 2006). To improve this, mobile robots with manipulation arms, which have the capability of moving around within the manufacturing environment and are not fixed to one physical location, could be employed. Moreover, it is possible for mobile robots to adapt to changing environments and perform a variety of tasks. Therefore, mobile robot technology holds great potential in the manufacturing industries (EUROP, 2009).

Despite considerable attention within the manufacturing domain, real-world implementations of mobile robots have been limited (Stopp et al., 2003; Katz et al., 2006; Datta et al., 2008; Hentout et al., 2010) although the needs for transformable and flexible automation are present (Hvilshøj and Bøgh, 2011). In addition, the problem of planning and scheduling the tasks of mobile robots in the real-world industrial applications has received little attention in the literature (Maimon et al., 2000; Hurink and Knust, 2002; Zacharia and Aspragathos, 2005). Although, this problem could be modelled in some respects comparable to the Travelling Salesman Problems (TSP) which belongs to the class of NP-hard combinatorial optimization problems, the surveyed approaches are not well suited and cannot be directly used to solve this problem. Therefore, new initiatives are required to realize industrial acceptance and maturation of the autonomous industrial mobile robots.

In general, the mobile robot technology finds most suitable applications within the logistic area (transportation and part feeding). The logistics tasks are either very suited for mobile robots (easily manageable work pieces) or out of scope (unmanageable work pieces, e.g. due to large size and/or weight). In contrast, pre-assembly, inspection and process execution tasks are generally not suited for the current state of mobile robots. Often, these tasks are tailored for human operators and not suited for automation, because they require dexterous manipulation skills and/or experience. Other application categories look promising, e.g. machine tending and cleaning but they are either few in numbers or low in suitability score. In addition, some application categories are generally out of scope, e.g. maintenance, repair and overhaul, as these tasks require handling of large parts (e.g. injection moulds) or task execution in inaccessible areas (e.g. the backside of manufacturing equipment). In summary, mobile robot technology, at its current stage, finds its most suitable applications within the logistic area, moves towards assistive tasks, and in the future more service and other non-production-related tasks.

Overall, the necessary mobile robot technologies (hardware and software) exist at a mature level. One significant reason that no autonomous mobile robots have yet been implemented in industrial environments is that research in the right applications have not been carried out. In this paper, the mobile robot technology is transferred from laboratory experiments to a real-world environment and application by proposing an approach consisting of:

- a mobile robot system design (Little Helper),
- an appropriate and comprehensive industrial application (multiple-part feeding),
- an implementation concept for industrial environments (the bartender concept), and
- a real-time heuristics to schedule the mobile robot in the industrial application.

The paper is organized as follows. Section 2 presents a methodology supporting the mobile robot technology within real-world manufacturing environments. Section 3 describes the necessary preparation and customization steps for the multiple-part feeding implementation at a company which produces pumps. Section 4 presents results from the conducted implementation in term of system performances. Finally, the paper concludes and discusses options for future work within the field of autonomous industrial mobile robots in Section 5.

2. METHODOLOGY

2.1 Little Helper

The autonomous industrial mobile robot "Little Helper" was developed by Hvilshøj and Bøgh (2011). The overall vision is shown in Fig. 1, which depicts a typical work schedule. "Little Helper" can be used for various manufacturing tasks, e.g. carrying tasks, preparatory and/or post-processing tasks, and even pre-assembly, quality inspection, and machine tending. Furthermore, the mobile robot is able to work fully automatic in a full shift of a working day. In this way, "Little Helper" becomes a flexible automation technology, which contributes to the realization of transformable production systems.

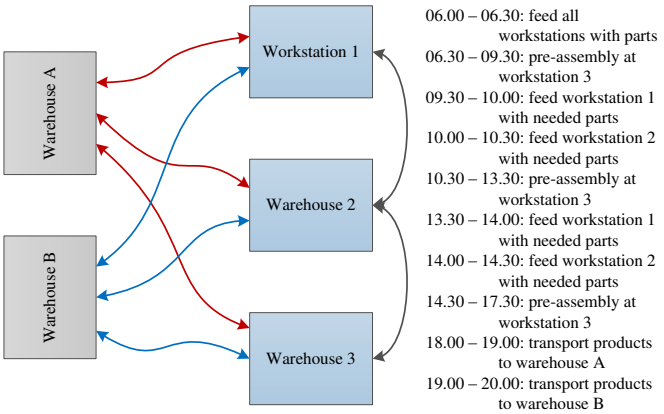

Fig. 1. A typical work day of a mobile robot.

The "Little Helper" system design shown in Fig. 2 relies on standardized, industrially accepted and commercial-off-the-shelf (COTS) hardware components and software packages, which offer significant savings in procurement, development and maintenance. In addition, "Little Helper" depends on mass customization principles, in terms of modularization, product families and product platforms. The architecture consists of different modules (basic building blocks) such as mobile platform, vision, robot manipulator, and tooling with pre-defined interfaces and skills/actions. In this way, it is possible to configure the mobile robot for versatile industrial applications and/or environments.

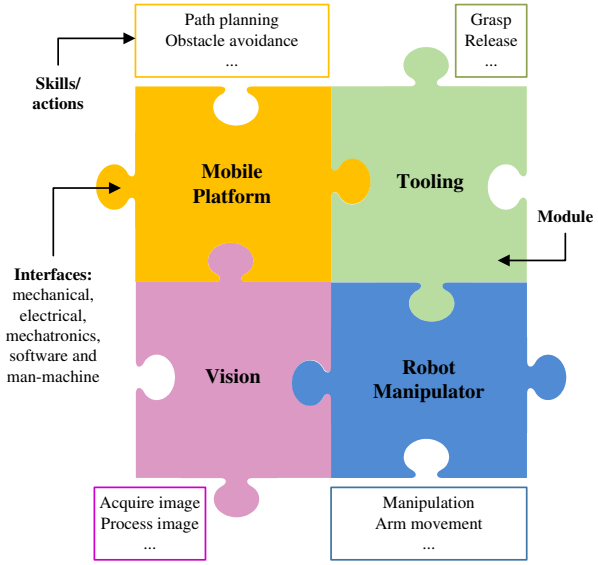

Fig. 2. A modular autonomous industrial mobile robot system (the first "Little Helper" prototype).

2.2 Multiple-part feeding

Currently, the most suitable process for the mobile robot is multiple-part feeding which is the task of loading several parts or components at a time into feeders (e.g. step feeders) and/or machines (e.g. turning-milling machines). Typical manufacturing facilities consist of several automatic production lines, where the feeding of parts or components is manually performed. Multiple-part feeding is a non-value adding manufacturing task and it is quite often disruptive (in-between and periodic) for the factory workers. Therefore, automation of multiple-part feeding holds great industrial potential. Utilizing autonomous mobile robots will make multiple-part feeding tasks more flexible and complete.

2.3 The bartender concept

Multiple-part feeding task is to some extend similar to material handling: pickup ⟶ transportation ⟶ empty at feeder ⟶ transportation ⟶ place. At the same time they differ in other aspects, e.g. the parts to be handled, the feeders to be serviced, and the workstations to be localized. To address this, the bartender concept is proposed for solving multiple-part feeding tasks by mobile robots (Fig. 3).

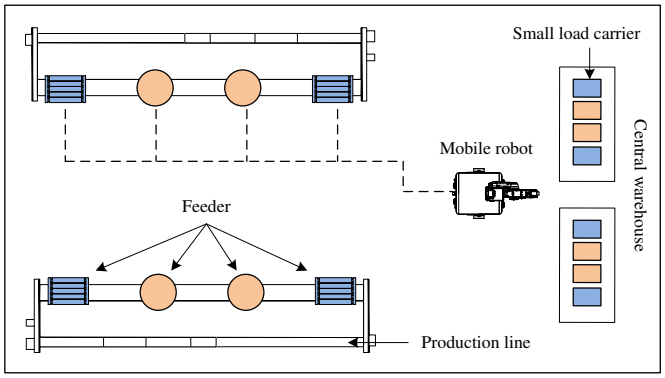

Fig. 3. Illustration of the bartender concept.

In the bartender concept, a central warehouse (a bar) is created to gather different needed parts into one area. An operator (the bartender) puts parts into small load carriers (SLCs) which are placed in the warehouse. The use of SLCs simplifies the manipulation tasks as different parts can be handled by the same robot tool. The SLCs are equipped with QR codes (used for identification and pose estimation) corresponding to the product numbers, which assist the robot to pick up the right parts. When a production line needs parts, the robot is requested via a wireless network, and then it retrieves and carries SLCs containing parts from the central warehouse, moves to feeder, empties all parts inside SLCs, and returns to the warehouse to unload all empty SLCs. The schedule of multiple-part feeding tasks of the robot is based on features of the feeders (maximum, minimum levels of parts, and part-feeding rate to production line) and data from the Manufacturing Execution System (MES).

The bartender concept is not a direct competitor to dedicated material handling systems, like the use of Kanban together with conveyor belts and AGV's, KIVA systems, etc. The bartender concept must be understood as part of the general mobile robot concept where the robot is able to perform a wide variety of manufacturing tasks. In the context of multiple-part feeding, the mobile robot is capable of performing other tasks, while waiting between the periodic feeding tasks, e.g. quality control, pre-assembly, machine tending, etc. This contributes to the realization of a flexible and transformable production system.

2.4 Real-time heuristics

In the bartender concept, to accomplish all the movements with the smallest consumed mount of battery energy, the total travelling time of the mobile robot is a critical objective. Apart from that, another performance measure is the amount of time feeders have been waiting to be replenished by the robot. In addition, making decisions on which sequence the robot should provide parts to feeders is part of the real-time operations of production planners. Therefore, in this section, genetic algorithm (GA), a random search approach taking over the principle of biological evolution (Goldberg, 1989), is applied to develop a real-time heuristics which converts the problem of scheduling multiple-part feeding tasks of the mobile robot in a manner that ensures best solutions can be found. Let $P(t)$ and $C(t)$ be parents and offspring in the generation t, respectively. The general implementation structure of the GA-based real-time heuristics is described in Fig. 4 as follows:

```
procedure: GA-based heuristic
input: system data (planning horizon, feeder features)
       robot data (working time, travelling time, max SLC)
       GA parameters
output: the best sequences of feeding tasks
begin
  t ← 0;
  initialize P(t) by random initialization routine;
  check the feasibility and repairing by constraint handling routine;
  evaluate P(t) by fitness assignment routine;
  while (not terminating condition) do
    create C(t) from P(t) by order crossover and insertion mutation routines;
    check the feasibility and repairing by constraint handling routine;
    evaluate C(t) by fitness assignment routine;
    select P(t + 1) from P(t) and C(t) by (μ + λ) selection routine;
    t ← t + 1
  end
end
```

Fig. 4. Structure of GA-based real-time heuristic.

- *Genetic representation*: a solution can be represented by a chromosome of non-negative integers $(0, i, j, \ldots, i, \ldots, k)$ which is an ordering of multiple-part feeding tasks of the robot where i, j, k: feeder index; $i, j, k \in \{1, 2, \ldots, n\}$; n: number of feeders. The original length of a chromosome is equal to the total number of requests of tasks (Σn_i; n_i: number of requests of task i).

- *Initialization*: for the initial generation, genes on a chromosome are randomly filled with tasks at feeders. The frequency of a task is the number of requests of that task, in other words, the number of times that tasks has to be executed (Dang et al., 2012).

- *Constraint handling*: after initialization or crossover and mutation routines, chromosomes are handled or repaired to be valid. A valid chromosome should satisfy two constraints on the limitation on carrying capacity Q_m of the robot and time windows of requests of the feeding tasks. For the first type of constraints, to guarantee the robot serves no more feeders than SLCs carried in one route, the task at the warehouse represented by zeroes (0) are inserted into a chromosome. The second type of constraints requires a request of a feeding task to start after release time and completed by the due time of that request, if possible. However, due time constraints are considered soft constraints and are thus modelled as an objective of the total tardiness of feeding tasks (which is shown in *fitness assignment*). The release and due time are determined as shown in Equation (1) and (2) below, while the constraint handling procedure is shown in Fig. 5.

$$r_{ik} = s_{ik\text{-}1} + (u_i - v_i)c_i,\ i \in \{1, 2, \ldots, n\};\ k \in \{1, 2, \ldots, n_i\} \quad (1)$$

$$d_{ik} = r_{ik} + v_i c_i,\ i \in \{1, 2, \ldots, n\};\ k \in \{1, 2, \ldots, n_i\} \quad (2)$$

where r_{ik}, d_{ik}, s_{ik}: release time, due time, starting time of request k of task i

u_i, v_i, c_i: maximum level, minimum level of parts of feeder i, and part-feeding rate to production line of feeder i

```
procedure: constraint handling
input: offspring C, max SLC Qm, time windows
output: repaired offspring
begin
  Q ← 0; // Q: the number of SLCs carried by the robot in a route
  p ← 0; // p: position of a task in a chromosome
  insert the first task 0 into C at position p;
  repeat
    repeat
      check the feasibility of time windows of next Qm tasks;
      if (time window not satisfied) then
        swap with tasks having greater due times;
      end
    until (time windows satisfied) or (time windows not satisfied with all
                                       possible swaps)
    if (time window satisfied) then
      Q ← Qm;
      insert another task 0 into C at position (p + Q + 1);
    elseif (time window not satisfied with all possible swaps) then
      if Q > 0 then
        Q ← Q – 1;
        reinsert the current task 0 into C at position (p + Q + 1);
      else
        select the reparation having the least total tardiness;
        Q ← Qm;
        insert another task 0 into C at position (p + Q + 1);
      end
    end
  until (p = length of C)
end
```

Fig. 5. Constraint handling procedure.

- *Fitness assignment*: after constraint handling procedure, the objectives of the total travelling time of the robot and the total tardiness of part-feeding tasks are calculated one after another for every chromosome in the population. A weighted-sum fitness function F is then used to assign a fitness value to each chromosome as shown in Equation (3) where t_{ij} is travelling time of the mobile robot from one location to another, and α is the weighted coefficient.

$$F = \alpha(\Sigma_{i,j}t_{ij}) + (1 - \alpha)(\Sigma_{i,k}(\max\{0, s_{ik} - d_{ik}\})) \quad (3)$$

- *Genetic operators*: selection, crossover, and mutation are three main genetic operators. For selection, the $(\mu + \lambda)$ selection method is used to choose chromosomes for reproduction. This selection mechanism guarantees that the best solutions up to now are always in the parent generation. For crossover, order crossover (OX) (Gen and Lin, 2008) operated with probability P_c will be employed to generate an offspring as follows. Genes having zero values are removed before two cut points are randomly chosen on the parent chromosomes. A string between these cut points in one of the parent chromosomes is first copied to the offspring, the remaining positions are then filled according to the sequence of genes in the other parent starting after the second cut point. When an offspring is produced, it undergoes insertion mutation (Gen and Lin, 2008) with probability P_m which selects a gene at random and inserts it in a random position.

- *Termination criteria*: termination criteria are used to determine when the GA-based real-time heuristic should be stopped. On the one hand, if the best solutions over generations do not converge to a value, the maximum generation G_m would be used to stop the run. On the other hand, if the best solution does not improve over G_c consecutive generations, it would not be valuable to continue searching.

2.5 Mission planning and control

An important part of the mobile robot technology is the ability to interact with industrial environments in a flexible manner. Therefore, the mobile robot must be able to communicate with the manufacturing equipment (machine-machine) and the operators (human-machine). By integrating the mobile robot into the general enterprise network, it is possible to plan and control globally, as the mobile robot becomes a resource on the same level as corresponding manufacturing device. In the bartender concept, this is handled by the Mission Planning and Control (MPC) software. Fig. 6 below depicts the overall structure of the MPC.

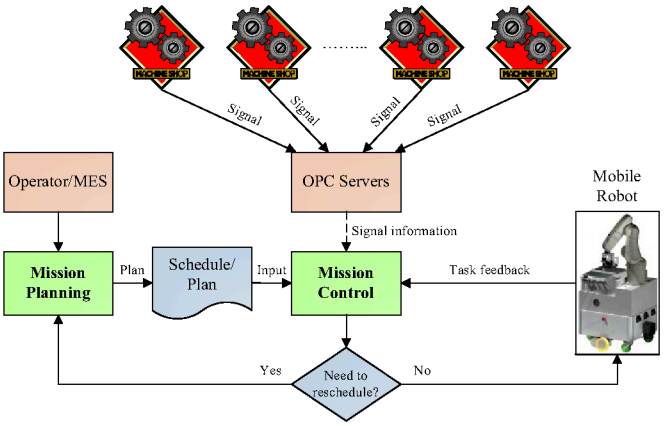

Fig. 6. Overall structure of Mission Planning and Control.

The MPC consists of two main components; Mission Planning and Mission Control. Mission Planning, into which the aforementioned real-time heuristic approach is integrated, is responsible to quickly generate best sequences of tasks under consideration of time window and capacity constraints, while Mission Control has the capability of transforming the generated plans into a real-time schedule based on the feedback from the mobile robot. At the first stage, Mission Planning will create an initial schedule for the mobile robot based on input data from the MES (e.g. demand, cycle time of products) and users (e.g. number of feeders, the limitation on carrying capacities, travelling time, and working time of the mobile robot). This schedule serves as an input to Mission Control in the next stage. The tasks specified in the initial plan is executed in order and controlled based on the feedback from the mobile robot, so that feeders will never run low on parts. However, during real-time operations, Mission Control may receive signals including feeder and part information that are required to be served because of different industrial disruptions. Such signals take priority over other planned tasks. Re-planning mechanism will then be activated immediately to update starting and finishing time(s) of the remaining tasks. After servicing the prioritized tasks, the MPC continues according to the updated schedule, until it receives new signals from OPC servers.

3. PREPARATION AND CUSTOMIZATION

3.1 The CR 1-2-3 impeller line

The chosen area for the real-world implementation is the CR 1-2-3 impeller production line that manufactures impellers for the CR pumps. The CR line consists of a warehouse and four feeders that have to be served by the mobile robot. The warehouse is indexed 0 and the feeders are indexed from 1 to 4 (N = {0, 1, 2, 3, 4}) and named Back Plate, Van Feeder 1, Van Feeder 2, and Front Plate respectively. Furthermore, different feeders are filled by different kinds of parts, namely back plates for feeder 1, vanes for feeder 2 and 3, and front plates for feeder 4. To produce an impeller on the CR 1-2-3 line, these three types of parts, which consists of six vanes with three from feeder 2 and the other three from feeder 3, one front plate from feeder 4 and one back plate from feeder 1, are automatically assembled. Fig. 7 particularly illustrates the aforementioned production area where the real-world implementation has been carried out.

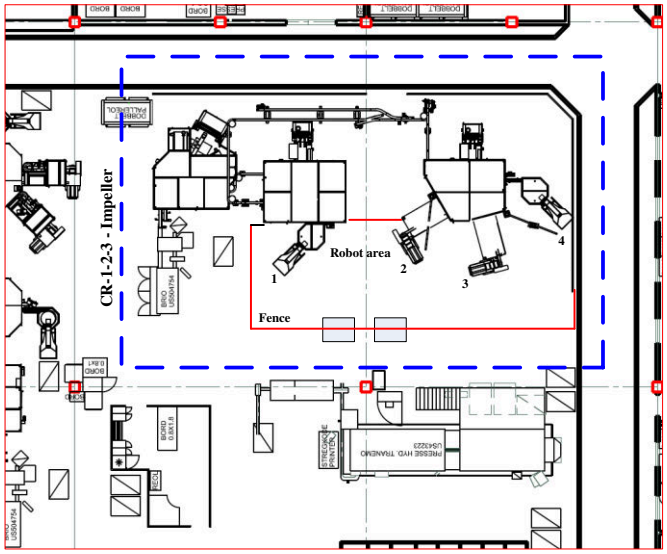

Fig. 7. The CR 1-2-3 impeller production line.

3.2 3D simulation and customization

Before testing the bartender concept at the CR 1-2-3 impeller line, 3D simulation and experiments were carried out. 3D simulations in DELMIA Robotics were used to virtually define and test manipulation, grasping strategies, and the transportation of SLCs. In addition, the 3D simulations were used for verifying that the "Little Helper" prototype is able to keep up with the production schedule and cycle time at the CR 1-2-3 impeller line. In general, the 3D simulations proved very effective in the design phase, especially as a link between academia and industry.

Based on the requirements from the CR1-2-3 impeller line, and the results from the 3D simulations and laboratory experiments, it is necessary to customize the mobile robot system and the industrial environment as shown in Fig. 8.

- *Customization of "Little Helper"*: a) *gripper*: compliant with the utilized SLCs; b) *SLC rack*: enable the robot to carry and transport several SLCs at a time; c) *vision*: implement standard algorithms for QR identification.
- *Customization of the industrial environment*: a) *safety*: use fences and warning signs to ensure that no people enter the area as well as to prevent the mobile robot leaving the area; b) *navigation and localization*: place reflector landmarks at the workstations; c) *warehouse*: customize the warehouse to have two tables with four slots on each table.

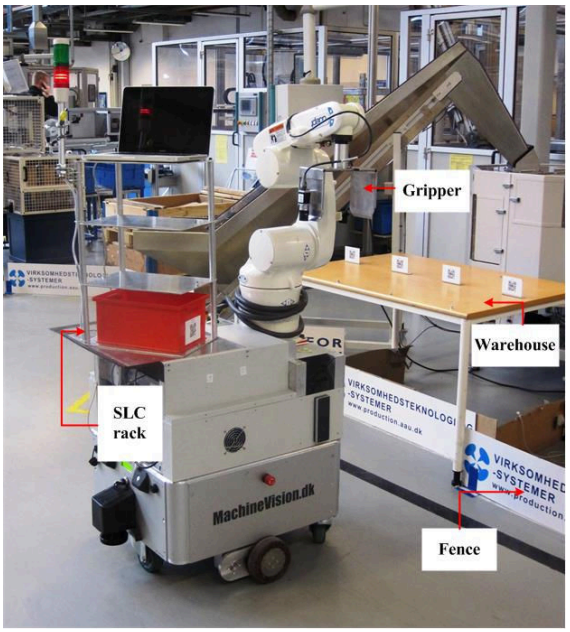

Fig. 8. Customization of "Little Helper" and the industrial environment.

4. REAL-WORLD IMPLEMENTATION

The system of the mobile robot and the MPC was tested for a period of three days at the CR 1-2-3 impeller production line. The overall goal was to acquire experience on what actually happens when the mobile robot is moved outside the laboratory and into a real-world manufacturing environment. Based on this, it is possible to find guidelines for future improvements within the system.

Overall, the test with "Little Helper" and the MPC software was a success. The longest successive execution time was about 45 minutes of continuous part feeding. "Little Helper" was able to continuously pick and place SLCs from/to the warehouse and empty them into different feeders of the CR 1-2-3 impeller line based on the best schedule generated from the proposed real-time heuristics of Mission Planning and signal information from the OPC servers. The best solution, depicted by the Gantt chart in Fig. 9, is given: 0 - 4 - 1 - 0 - 4 - 1 - 0 - 4 - 3 - 0 - 1 - 1 - 0 - 4 - 2 - 0, with total travelling time of 10.7 minutes and no tardiness in part feeding. The best solution was obtained within a second; hence, the proposed real-time heuristics also demonstrated its capability of finding the best solution in online production mode.

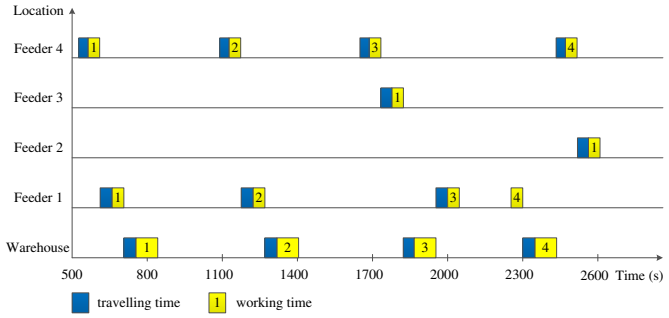

Fig. 9. Gantt chart for the best solution of the test.

In this real-world implementation, a load of 59 per cent was put on the "Little Helper" prototype. About 40 per cent of the run time was spent on moving the mobile platform between feeders or travelling time (75 per cent of maximum speed and acceleration), while the remainder was spent on moving the robot manipulator or working time (60 per cent of maximum speed and acceleration). Owing to safety and robustness issues, the test was not executed with maximum speeds and acceleration on the various subsystems of the mobile robot. The other performance characteristics of the mobile robot are listed as follows: a) mobile platform accuracy: ± 10 mm; b) effective platform payload: 50 kg; c) manipulation accuracy (vision guided): ± 1 mm; d) effective manipulator payload: 2 kg; e) continuous runtime (battery): 4 hours.

5. CONCLUSION AND FUTURE WORK

This paper investigates the potential of mobile robots based on a COTS system design, a general implementation concept for multiple-part feeding (the bartender concept), and a real-time heuristics integration into the MPC to schedule mobile robots in the industrial application. The system and concept have been evaluated in a real-world case study at an impeller production line with the mobile robot prototype "Little Helper". The main purpose is to acquire experience on what actually happens when a mobile robot is moved outside the laboratory and into a real-world manufacturing environment. Based on this, it is possible to find guidelines for future research and developments within mobile robots.

Overall, the real-world implementation at the impeller production line was a success, as the experiments showed that the mobile robot was capable of performing a meaningful industrial task. The system and concept only demand minor changes to the environment, most importantly putting parts into SLCs instead of pallets. Furthermore, the real-time heuristic approach proved valuable as the best schedule of the mobile robot could be found in a new environment within seconds. The performance and robustness of the system can in many ways be improved, e.g. by shifting towards an integrated framework and/or improving the interaction between the subsystems (modules).

Based on the experience gained from the conducted test at the manufacturing environment, the immediate challenges that need to be solved for mobile robots to become a feasible and mature technology are: increase of performance (efficiency) and robustness (error recovery) of the total system (robot and MPC), increase of the system capability (e.g. handling of more complex manufacturing tasks), and increase of system safety (human-machine-interaction).

For future work, "Little Helper" is being improved to perform more advanced tasks such as pre-assembly, quality inspection and machine tending, while the MPC considers the problem of planning and scheduling a fleet of mobile robots in the manufacturing environment of these advanced tasks.

Acknowledgements. This work has partly been supported by the EC under 260026-TAPAS.

REFERENCES

Dang, Q.V., Nielsen, I., and Bocewicz, G. (2012). A genetic algorithm-based heuristic for part-feeding mobile robot scheduling problem. In J.M.C. Rodríguez (ed.), *Trends in PAAMS*, *AISC 157*, 85–92. Springer Verlag, Berlin.

Datta, S., and Ray, R. (2007). AMR vision system for perception, job detection and identification in manufacturing, *Vision Systems: Applications*, *InTech*, doi: 10.5772/5004.

EUROP (2009). *The Strategic Research Agenda for Robotics in Europe – Robotics Visions to 2020 and Beyond*. EUROP, Brussels.

Gen, M., and Lin, L. (2008). *Network Models and Optimization*. Springer Verlag, London.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, New York.

Hentout, A., Bouzouia, B., Akli, I., and Toumi, R. (2010). Mobile manipulation: a case study, *Robot Manipulators*: *New Achievements*, *InTech*, doi: 10.5772/9324.

High Level Group (2006). *Manufacture*: *Strategic Research Agenda*. High Level Group, Luxembourg.

Hurink, J., and Knust, S. (2002). A tabu search algorithm for scheduling a single robot in a job-shop environment, *Discrete Applied Mathematics*, 119 (1–2), 181–203.

Hvilshøj, M., and Bøgh, S. (2011). "Little Helper" – an autonomous industrial mobile manipulator concept, *Int. Journal of Advanced Robotic Systems*, 8(2), 80–90.

Jovan, F., Koren, Y., and Boër, C.R. (2003). Present and future of flexible automation: towards new paradigms, *CIRP Annuals – Manufacturing Technology*, 52 (2), 543–560.

Katz, D., Horrell, E., Yang, Y., et al. (2006). The UMass Mobile Manipulator UMan: an experimental platform for autonomous mobile manipulation. *Proceedings* of *Workshop on Manipulation in Human Environments at Robotics*: *Science and Systems*, Philadelphia, PA.

Maimon, O., Braha, D., and Seth, V. (2000). A neural network approach for a robot task sequencing problem, *Artificial Intelligence in Engineering*, 14 (2), 175–189.

Stopp, A., Horstmann, S., Kristensen, S., and Lohnert, F. (2003). Towards interactive learning for manufacturing assistants, *IEEE Transactions on Industrial Electronics*, 50 (4), 705–707.

Zacharia, P.T., and Aspragathos, N.A. (2005). Optimal robot task scheduling based on genetic algorithms, *Robotics and Computer-Integrated Manufacturing*, 21(1), 67–79.

Paper I

A methodology for implementation of mobile robot in industrial application

This paper has been submitted to Robotics & Computer-Integrated Manufacturing

A Methodology for Implementation of Mobile Robot in Industrial Application

Quang-Vinh Dang, Izabela Nielsen

Department of Mechanical and Manufacturing Engineering, Aalborg University, Fibigerstræde 16, DK 9220 Aalborg East, Denmark

vinhise@m-tech.aau.dk, izabela@m-tech.aau.dk

Abstract. The paper deals with the problem of the implementation of an autonomous industrial mobile robot in real-world industrial applications. The paper proposes a methodology for implementation consisting of: a mobile robot system design (Little Helper prototype), an appropriate industrial application (multiple-part feeding), an implementation concept for the industrial application (the Bartender Concept), a mathematical model, and a genetic algorithm-based heuristic. In the methodology, the mathematical model is proposed to formulate and find optimal solutions for the mobile robot scheduling problem which is described based on the implementation concept. This mathematical method is however only applicable to small-scale problems. A genetic algorithm-based heuristic is then proposed to find near-optimal solutions. A real-world demonstration at an impeller production line in a real factory and computational experiments are conducted to demonstrate the effectiveness of the proposed methodology.

Keyword: robotics; industrial application; scheduling; heuristics

1 Introduction

Today's manufacturing system ranges from mass production to customized production. The former has efficiency in high volumes but lacks flexibility. The latter however has opposite characteristics. Therefore, there is a need for a transformable system that combines the advantages of these two by using assistive automation and robotics. In industry, robots have been widely used to perform 4D tasks, namely dumb, dangerous, dull, and/or dirty. Industrial robotics thus becomes an essential part of the manufacturing backbone. However, the robots are rather inflexible as they are often dedicated and fixed [13]. One way to improve this is the use of mobile robot, an interdisciplinary technology extending the prospective application of industrial robotics by combining locomotion capabilities with manipulation abilities. Task flexibility and robotic mobility are two main advantages that mobile robots bring to industrial applications. Moreover, mobile robots are able to adapt to changing environments and perform

a variety of industrial tasks. The tasks include such processes as: transporting materials, machine tending, pre-assembly or quality inspection. Therefore, the mobile robot technology holds great potential in the manufacturing industries.

Mobile robot is a term used to refer to robotic systems including a robot arm mounted on a mobile platform, extended by a vision and tooling system, respectively. The concept of mobile robot first appeared in 1984 [24]. However, at that time its use and implementation were still limited due to high system costs and lack of processing power. Much research has been carried out since then, and until recently there have been many on-going mobile manipulation works within a variety of fields, e.g. professional service, domestic service, space, military, and industry. Autonomous industrial mobile robots, in particular, operate in structured environments. They however require a higher level of operational efficiency, e.g. in terms of speed, accuracy, robustness, and task planning and scheduling to be suitable for industrial applications. Hence, mobile robots combine the flexibility of service robots with the efficiency of industrial robots.

Despite the fact that manufacturers express a need for transformable and flexible automation [21], the real-world implementations of mobile robots have been limited [7,12,27]. Furthermore, the necessary technology components are, to a large extent, commercial off-the-shelf (COTS) components. In the field of industrial mobile robots, the major focus has been driven to the optimization of the individual technologies, especially robot motion, perception, and grasping of objects [17], while little attention has been paid to the use and application of these robots. Therefore, new initiatives are required in order to realize industrial acceptance and maturation of mobile robots.

In general, the mobile robot technology finds most suitable applications within the logistic area. The logistics tasks are either very suited for mobile robots (easily manageable work pieces) or out of scope (unmanageable work pieces, e.g. due to large size and/or weight). In contrast, pre-assembly, inspection, and process execution tasks are generally not suited for the current state of mobile robots. These tasks are often tailored for human operators and not suited for automation, because they require dexterous manipulation skills and/or experience. Other application categories look promising, e.g. machine tending and cleaning but they are either short in numbers or low in suitability score. Moreover, some application categories are generally out of scope, e.g. maintenance, repair, and overhaul, as

these tasks require handling of large parts (e.g. injection moulds) or task execution in inaccessible areas (e.g. the backside of manufacturing equipment). In summary, the mobile robot technology at its current stage finds most appropriate applications within the logistic area, specifically transportation and part feeding. However, to utilize the mobile robot in an efficient manner requires the ability to properly scheduling part feeding tasks in relation to the needs of given production lines. Hence, it is important to schedule in which sequences the mobile robot processes part feeding tasks so that it could effectively work while satisfying a number of practical constraints.

The problem of scheduling part feeding tasks of the mobile robot has been modelled in several respects comparable to the Asymmetric Travelling Salesman Problem (ATSP) which belongs to the class of NP-hard combinatorial optimization problem [10]. However, the problem is not identical to the ATSP because of some additional constraints which the problem possesses. Several approaches and models for exact or heuristic algorithms have been proposed to address problem of this type. Carpaneto et al. [2], Turkensteen et al. [28], Germs et al. [10] present branch-and-bound algorithm for solving the ATSP. Edan et al. [8] introduce a near-minimum task-planning algorithm for a fruit harvesting robot to find near-optimal-time path between the N given fruit locations. It is shown to be a solvable case of the Travelling Salesman Problem (TSP). Ascheuer et al. [1] present a cutting plane approach to the sequential ordering problem similar to the robot task-sequencing problem and finds minimum cost paths subject to precedence constraints. Dang et al. [4] propose a mixed-integer programming (MIP) model to obtain the optimal feeding sequence of a mobile robot. However, the performance of the MIP model is not evaluated and compared with other methods. For small task-scheduling problem, the aforementioned techniques can be used to find the optimal solutions of the problems. Nevertheless, they tend to get computationally intractable for large and complex problems [20]. Larger problems call for heuristic solutions. The heuristic approaches that are frequently applied to TSP, ATSP, or robot task-scheduling problems include: simulated annealing [9,23], tabu search [14], neural network [20], and genetic algorithm (GA) [3,19,22,26,29]. Zacharia and Aspragathos [30] introduce methods based on GAs and innovative encoding to determine the optimal sequence of robot's task points which is considered as an extension to the TSP.

Although much related research has been completed, the problem of scheduling a mobile robot with time windows, restricted capacity, and multiple tasks carried out during a planning horizon has received little attention in the literature despite its important application, e.g. part feeding tasks. In this problem, a number of tasks with time windows should be satisfied by the mobile robot if possible. A soft time window scenario allowing violating the upper bound constraints will be considered. Moreover, due to the limit on carrying capacity, after satisfying some tasks, the mobile robot has to return to a warehouse/based to load parts so that it can serve other tasks in the next route and so on (a route is from the warehouse, to locations of tasks, and back to the warehouse). The objectives of minimizing the total travelling time of the robot and the total tardiness of tasks are taken into account to support the global objective of maximizing system throughput. The surveyed approaches are not well suited and cannot be directly used to solve this problem due to the lack of a mechanism handling both soft time window and capacity constraints in case of the single robot. In the previous work, Dang et al. [6] present an MIP model and a heuristic based on GA to solve the problem of sequencing feeding tasks. However, these methods are developed to deal with the single objective function and hard time window scenario. Furthermore, the heuristic in [6] may be unable to find feasible solutions due to the mechanism of replacing infeasible chromosomes (solutions) with randomly generated ones. In this paper, a mathematical model with the determination of time windows and a GA-based heuristic with a suitable and advanced constraint handling operator are developed in order to particularly solve the mobile robot scheduling problem.

Overall, the necessary mobile robot technologies (hardware and software) exist at a mature level. The reason that no autonomous mobile robots have yet been implemented in industrial environments is that research in the right applications have not been carried out. In this paper, the mobile robot technology is transferred from laboratory experiments to real-world environment and applications by proposing a methodology consisting of:

- a mobile robot system design (Little Helper),
- an appropriate industrial application (multiple-part feeding),
- an implementation concept for industrial environments (the Bartender Concept) and the concept-based scheduling problem description,

- a mathematical model to formulate the scheduling problem and find the optimal schedule of multiple-part feeding tasks of the mobile robot,
- a genetic-algorithm based heuristics to schedule multiple-part feeding tasks of the mobile robot in the industrial application.

The paper is organized as follows. Section 2 presents a methodology supporting the mobile robot technology within real-world manufacturing environments. Section 3 describes the preparation steps for a multiple-part feeding demonstration at a company which produces pumps. Section 4 illustrates the results of the demonstration using real-world data from a Danish production company, from the genetic-algorithm based heuristic and compares its performance with that of the mathematical model. Computational experiments are also conducted in this section. Finally, the paper concludes and discusses options for future works in Section 5.

2 Methodology

2.1 Mobile robot system design

The autonomous industrial mobile robot, Little Helper, was modelled and developed by Hvilshøj and Bøgh [15]. Little Helper can be used for various manufacturing tasks, e.g. carrying, preparatory, and post-processing tasks, even pre-assembly, quality inspection, and machine tending. Furthermore, the mobile robot is able to work fully automatic in a third shift of a working day [16]. In this way, Little Helper becomes a flexible automation technology, which contributes to the realization of transformable production systems.

The Little Helper system design shown in Fig. 1 relies on standardized, industrially accepted, and COTS hardware components and software packages, which offer significant savings in procurement, development, and maintenance. In addition, Little Helper depends on mass customization principles, in terms of modularization, product families, and product platforms. The architecture consists of different modules (basic building blocks) such as mobile platform, vision, robot manipulator, and tooling with pre-defined interfaces and skills/actions. In this way, it is possible to configure mobile robot for versatile industrial applications and/or environments.

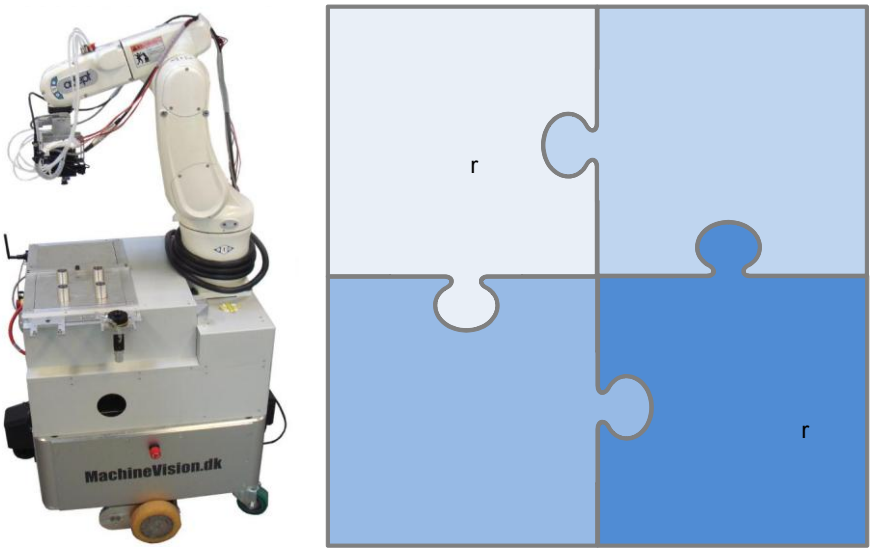

Fig. 1 A modular mobile robot system

2.2 Industrial application

Currently, the most suitable process for the mobile robot is multiple-part feeding which is the task of loading several parts or components at a time into feeders (e.g. step feeders). These feeders are designed to automatically supply parts to machines of one or several production line(s) in a typical manufacturing cell. Multiple-part feeding is manually performed, non-value adding manufacturing task and quite often disruptive (in-between or periodic) for production workers. Furthermore, when the workers forget to fill the feeders, this may lead to stopping the production lines. Therefore, automation of multiple-part feeding holds great industrial potential. Utilization of the mobile robot instead of humans can reduce the dependence on human intervention and make multiple-part feeding tasks more flexible and complete.

2.3 Implementation concept and scheduling problem description

Multiple-part feeding task has many similarities to material handling (pickup, transportation, empty at feeder, transportation, and place), but at the same time they differ in other aspects, e.g. the parts to be handled, the feeders to be serviced and the workstations to be localized. To address these, the Bartender Concept is proposed for solving multiple-part feeding tasks by the mobile robot (see Fig. 2).

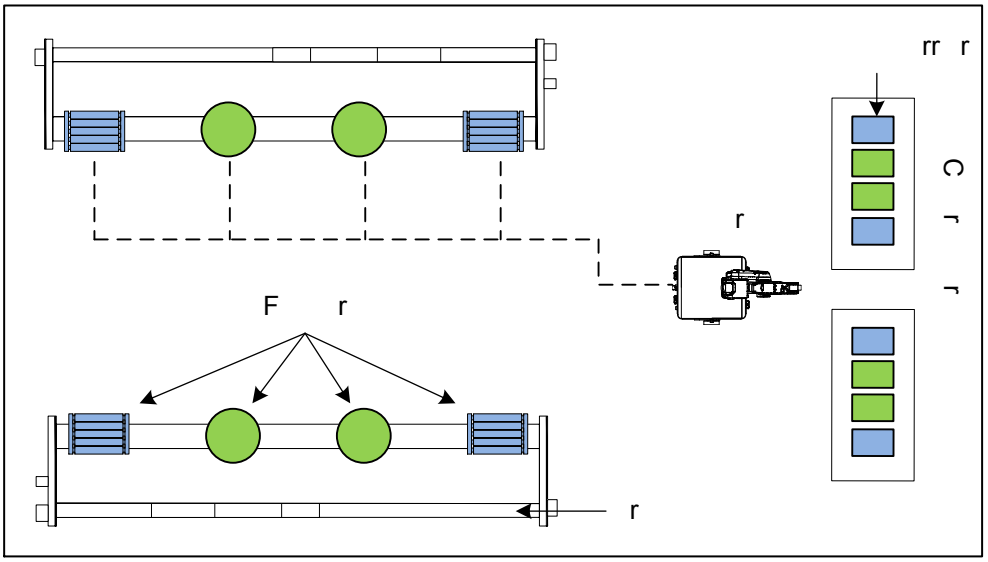

Fig. 2 Illustration of the Bartender Concept

In this concept every feeder is assigned the following features: maximum level, minimum level, and part-feeding rate to machine. Furthermore, a central warehouse (a bar) is created to gather different parts into one area. An operator (the bartender) puts parts into small load carriers (SLCs) which are placed in the warehouse. The number of parts inside each SLC is equal to the difference between maximum level and minimum level of parts of the feeder in which that SLC is emptied. The use of SLCs simplifies the tasks as different parts can be handled by the same robot tool. The SLCs are equipped with QR codes (used for identification and pose estimation) corresponding to the product numbers which assist the mobile robot to pick up the right parts. When the mobile robot is requested via a wireless network, it will retrieve and carry one or several SLCs containing parts from the warehouse, move to feeder locations, empty all parts inside SLCs, and return to the warehouse to unload all empty SLCs. Each feeder has to be served a number of times in order to keep the production line in operation. The mobile robot thus has a set of requests possessing time windows to carry out for each feeder during the planning horizon. In order to accomplish all the movements with the smallest consumed amount of battery energy and thereby increase the availability of the mobile robot, the total travelling time of the mobile robot must be taken into account. Furthermore, the mobile robot is allowed to start processing a request at a feeder after the upper bound of the time window of that request. Hence, another important performance measure is the amount of time feeders have been waiting to be replenished by the mobile robot, i.e. the tardiness

of tasks. Note that making decisions on which sequences the mobile robot should provide parts to feeders is a part of real-time operations of production planners. It means that the best solution must be quickly obtained at the beginning of production shifts or during the shifts due to errors in a manufacturing cell (e.g. machine breakdown) or changes in a manufacturing cell's conditions (e.g. cycle time of production lines). Moreover, as the problem is NP-hard, computation time exponentially grows with the size of the problem (e.g. longer planning horizon, larger number of feeders). It is therefore necessary to develop a computationally effective algorithm, namely a GA-based heuristic, to sequence the tasks in order to minimize a weighted objective of the total travelling time of the mobile robot and the total tardiness of the tasks while satisfying a number of practical constraints. To evaluate the performance of the proposed heuristic, an MIP model is formulated in the next section.

2.4 Mathematical model

In this section, an MIP model is developed to determine an optimal sequence in which the mobile robot visits n feeders to process multiple-part feeding tasks. All feeding tasks, corresponding to deliveries of SLCs, are known in advance. A method based on the (s, Q) inventory system [25], to protect against shortage of parts over a replenishment lead time, is also presented to determine the time windows of the multiple-part feeding tasks. A soft time windows scenario allowing violating the upper bound will be taken into account in this model. The mobile robot is based at a central warehouse, and a limit on the carrying capacity of the mobile robot is imposed. The MIP model contains a number of decision variables that are constrained to have only integer values. Integer variables make optimization problems non-convex and thus far more difficult to solve. Memory and solution time may rise exponentially as the size of problem increases with more added integer variables. Therefore, in practice the MIP model could be applicable only to small-scale problems with few feeders on production line(s) and a short planning horizon, i.e. few tasks to schedule. For these scenarios, the MIP model will give optimal solutions which could be used as reference points to quantify the scale of benefits achieved by the GA-based heuristic (further developed in Section 2.5). Assumptions, notations, time windows, and the formulation of the MIP model are given in the following subsections.

2.4.1 Assumptions

- An autonomous industrial mobile robot is considered in a disturbance free environment.
- The mobile robot can carry one or several SLC(s) at a time.
- All tasks are periodic, independent, and assigned to the same robot.
- Working time and travelling time of the mobile robot and the part-feeding rate to the machine of a feeder are known and constant.
- All feeders of machines are fed up to the maximum levels, and the mobile robot starts from the warehouse at the initial stage.

2.4.2 Notation

N : set of all tasks ($N = \{0, 1, 2, \ldots, n\}$ where 0: task at the warehouse)

n_i : number of times task i has to be executed

R : set of all possible routes ($R = \{1, 2, \ldots, R_{max}\}$, $R_{max} = \Sigma\, n_i$, $\forall i \in N \setminus \{0\}$)

p_i : periodic time of task i

w_i : working time of robot at location of task i

t_{ij} : travelling time of robot from location of task i to location of task j

c_i : part-feeding rate to machine of feeder i

v_i : minimum level of parts in feeder i

u_i : maximum level of parts in feeder i

Q_m : maximum number of SLCs could be carried by robot

T : planning horizon

Decision variables:

$$x_{ik}^{jlr} = \begin{cases} 1 & \text{if robot travels from location of task } i \text{ (request } k\text{) to location of task } j \\ & \text{(request } l\text{) in route } r \\ 0 & \text{otherwise} \end{cases}$$

y_{ik} : route index to which request k of task i belongs

e_{ik} : release time of request k of task i

d_{ik} : due time of request k of task i

s_{ik} : starting time of request k of task i

2.4.3 Time windows

A (s, Q) inventory policy is used with the defined features of the feeders to determine the time windows of the multiple-part feeding tasks as shown in Fig. 3.

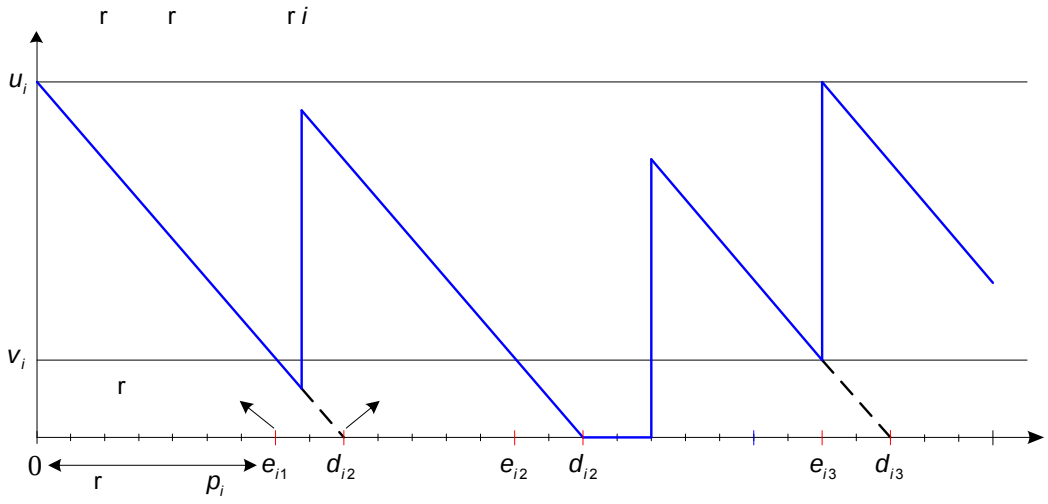

Fig. 3 Time windows of multiple-part feeding tasks based on the (*s*, *Q*) inventory system

Because of the periodic characteristic of the task for feeder *i* whose periodic time is calculated by $p_i = (u_i - v_i)c_i$, a number of requests must be carried out. The number of requests of task *i* is given by $n_i = \lfloor T / p_i \rfloor$. The mobile robot should start processing a request *k* of task *i* within the associated time window of that request if possible. It means that the mobile robot is allowed to arrive at feeder *i* after the upper bound of the time window (or the due time). In other words, the due time constraints are considered as soft constraints. Thus, these constraints can be modelled as an objective of the total tardiness of tasks. Note that a late arrival of the mobile robot at feeder *i* causes a shift in the time windows of the next requests of task *i*. If the mobile robot arrives at feeder *i* before the lower bound of a time window (or the release time), it will wait to begin service. The release time of request *k* of task *i* is set to the time when the number of parts inside feeder *i* drops to a certain level v_i. The due time of request *k* of task *i* is defined to the time when there are no parts in feeder *i*. The release time and due time of a request of a task are determined by Constraints (2) and (3) presented in the following MIP model.

2.4.4 Mixed-integer programming model

Objective function:

$$\min \left(\alpha \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N} \sum_{l=1}^{n_j} \sum_{r \in R} t_{ij} x_{ik}^{jlr} + (1-\alpha) \sum_{i \in N \setminus \{0\}} \sum_{k=1}^{n_i} \max(0, s_{ik} - d_{ik}) \right) \quad (1)$$

Subject to:

$$e_{ik} = e_{ik-1} + p_i + \max\left(0, s_{ik-1} - d_{ik-1}\right) \times 1_{\{k \geq 2\}} \quad \forall i \in N \setminus \{0\}, k \in \{1,2,...,n_i\}, e_{i0} = 0 \quad (2)$$

$$d_{ik} = e_{ik} + \left(v_i - 0\right) c_i \qquad \forall i \in N \setminus \{0\}, k \in \{1,2,...,n_i\} \quad (3)$$

$$s_{ik} \geq e_{ik} \qquad \forall i \in N \setminus \{0\}, k \in \{1,2,...,n_i\} \quad (4)$$

$$\sum_{j \in N \setminus \{0\}} \sum_{l=1}^{n_j} x_{01}^{jl1} = 1 \quad (5)$$

$$\sum_{j \in N \setminus \{0\}} \sum_{l=1}^{n_j} \sum_{r \in R} x_{01}^{jlr} \leq 1 \quad (6)$$

$$\sum_{(i,k),(j,l) \in Z} x_{ik}^{jlr} \leq \left|Z\right| \text{-1} \quad \forall r \in R, \forall Z \subseteq Z_T, Z_T = \{(i,k) \mid i \in N \setminus \{0\}, k \in \{1,2,...,n_i\}\} \quad (7)$$

$$\sum_{j \in N} \sum_{l=1}^{n_j} \sum_{r \in R} x_{ik}^{jlr} = 1 \qquad \forall i \in N \setminus \{0\}, k \in \{1,2,...,n_i\} \quad (8)$$

$$\sum_{i \in N} \sum_{k=1}^{n_i} \sum_{r \in R} x_{ik}^{jlr} = 1 \qquad \forall j \in N \setminus \{0\}, l \in \{1,2,...,n_j\} \quad (9)$$

$$\sum_{i \in N} \sum_{k=1}^{n_i} \sum_{j \in N \setminus \{0\}} \sum_{l=1}^{n_j} x_{ik}^{jlr} \leq Q_m \qquad \forall r \in R \quad (10)$$

$$s_{ik} + \left(w_i + t_{ij} \sum_{r \in R} x_{ik}^{jlr} \right) - L\left(1 - \sum_{r \in R} x_{ik}^{jlr}\right) + \left(y_{jl} - y_{ik}\right) \times \left(t_{i0} + w_0 + t_{0j} - t_{ij}\right) \leq s_{jl} \quad (11)$$

$$\forall i, j \in N, k \in \{1,2,...,n_i\}, l \in \{1,2,...,n_j\}$$

$$y_{jl} = \sum_{i \in N} \sum_{k=1}^{n_i} \sum_{r \in R} r \times x_{ik}^{jlr} \qquad \forall j \in N \setminus \{0\}, l \in \{1,2,...,n_j\} \quad (12)$$

$$y_{jl} \geq y_{ik} \sum_{r \in R} x_{ik}^{jlr} \qquad \forall i, j \in N, k \in \{1,2,...,n_i\}, l \in \{1,2,...,n_j\} \quad (13)$$

$$x_{ik}^{jlr} \in \{0,1\} \qquad \forall r \in R, \forall i, j \in N, k \in \{1,2,...,n_i\}, l \in \{1,2,...,n_j\} \quad (14)$$

y_{ik}: positive integer variable $\qquad \forall i \in N \setminus \{0\}, k \in \{1,2,...,n_i\}$ (15)

The objective function (1) minimizes a weighted objective of the total travelling time of the mobile robot and the total tardiness of tasks being sequenced, where α is the weighted coefficient. Constraints (2) and (3) determine the release time and due time of a request of a task respectively while constraint (4) guarantees that a request of a task is started after the release time of that request. Constraints (5) and (6) ensure that the mobile robot starts from the warehouse at the initial stage. Constraint (7) eliminates the sub-tours among requests of tasks, where Z is a subset of Z_T, where Z_T is a set of all requests of

tasks at feeders and the warehouse. Constraints (8) and (9) ensure that a request of a task is completed exactly once. Constraint (10) forbids the mobile robot to load a higher number of SLCs than its maximum capacity in the number of SLC Q_m. Constraint (11) handles the travelling time requirements between any pair of requests of tasks, where L is a given sufficiently large constant. In case two requests of the same task or different tasks are connected but are not in the same route, the mobile robot should visit the warehouse to unload empty SLCs and load filled ones. Constraint (12) assigns a request of a task to a route, and constraint (13) guarantees an ascending sequence of route indices for requests of tasks. Constraints (14) and (15) imply the types of variables.

2.5 Genetic algorithm-based heuristic

In this section a genetic algorithm, a search approach taking over the principle of biological evolution [11], is used to develop a heuristic. The GA-based heuristic allows converting the problem of multiple-part feeding tasks of the mobile robot so that near-optimal solutions can be found. Let $P(g)$ and $C(g)$ be parents and offspring in the generation g, respectively. The procedure of the GA-based heuristic shown in Fig. 4 is composed of the following main steps: genetic representation and initialization; constraint handling and fitness assignment; genetic operators including crossover, mutation and selection; termination criteria.

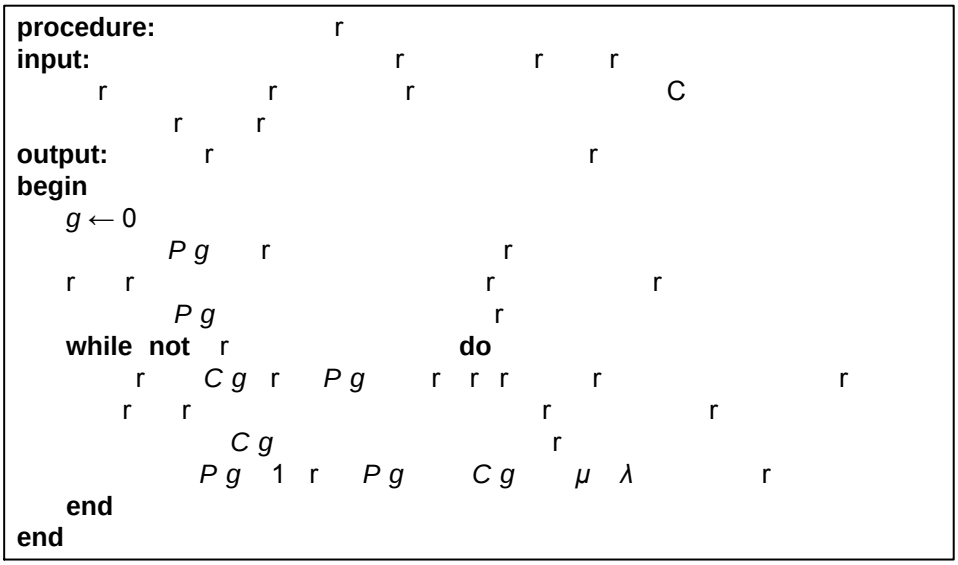

Fig. 4 Procedure of GA-based heuristic

2.5.1 Genetic representation and initialization

For the problem under consideration, the path representation is used to encode a

chromosome/solution which represents an ordering of requests of tasks of the mobile robot as follows: (0, i, j, …, i) where i, j: feeder/task index; $i, j \in \{1, 2, \ldots, n\}$. If a task is requested more than once, different genes on a chromosome may contain the same task index. The original length of a chromosome is equal to the total number of requests of all tasks added to the first request of the warehouse task ($1 + \Sigma n_i$).

For the initial generation, tasks are placed into genes on a chromosome at random. The frequency of a task is the number of requests of that task, in other words, the number of times that task has to be executed.

2.5.2 Constraint handling and fitness assignment

After the initialization or crossover and mutation operations, chromosomes are repaired to handle the two constraints: limit on the carrying capacity Q_m of the mobile robot and time windows of requests of tasks. The first constraints require that the mobile robot does not serve more requests of tasks than the number of SLCs it is carrying. The second constraints require a request of a task to be started within the release time and the due time of that request, if possible. As mentioned, due time constraints are considered as soft constraints. Thus, they can be modelled as an objective of the total tardiness of tasks in the fitness assignment. An approach of handling these constraints is developed and applied to each chromosome in the initial and descendant generations as shown in Fig. 5 along with the description below.

Step 1: Compute release times and due times of the genes on the chromosome containing the first requests of different tasks.

Step 2: Assign the maximum number of SLCs Q_m to the actual number of SLCs, denoted as Q, carried by the mobile robot in a route ($Q \leftarrow Q_m$).

Step 3: Compute starting times and check the satisfaction of time windows constraints for the next Q genes.

If starting time of any gene among the next Q genes is not satisfied its time window, then go to Step 4.

Otherwise, update release times and due times of the genes (on the chromosome) which contain the next requests of the tasks on these Q genes, then go to Step 9.

Step 4: Considering genes from position 1 to position p of a gene i which is not satisfied its time windows among the next Q genes, make a set D of positions of genes whose due times are greater than that of gene i.

Step 5: Insert gene i to a position p_d randomly selected from set D.

Step 6: Re-compute starting times and check the satisfaction of time windows constraints for the genes from the position p_d to position p.

If starting time of any gene from the position p_d to position p is not satisfied its time window, then check if all positions in set D are selected. If not, then go back to Step 5. If so, then go to Step 7.

Otherwise, update release times and due times of the genes (on the chromosome) which contain the next requests of the tasks on the genes from the position p_d to position p, then go to Step 9.

Step 7: Decrease Q by one unit ($Q \leftarrow Q - 1$) and check if Q is 0. If not, then go back to Step 3. If so, then go to Step 8.

Step 8: Select the value of Q and the insertion carried out in Step 5 which results in the minimum tardiness.

Step 9: Considering the gene (0) of the last warehouse task at position p_w on the chromosome, insert another gene (0) of the warehouse task at position $p_w \leftarrow p_w + Q + 1$. Note that, the chromosome length is increased by one unit after each gene (0) insertion.

Step 10: Determine if the constraint handling approach for the chromosome ends.

If chromosome length $- p_w > 0$, then check if chromosome length $- p_w \geq Q_m$. If so, $Q \leftarrow Q_m$. If not, $Q \leftarrow$ chromosome length $- p_w$. Then go back to Step 3.

Otherwise, end the constraint handling approach.

Following the approach of handling constraints, the fitness assignment is carried out. The objectives of the total travelling time of the mobile robot and the total tardiness of multiple-part feeding tasks are calculated one after another for every chromosome in the population. A weighted fitness function F is then used to assign a fitness value to each chromosome as: $F = \alpha\sum$(travelling time of the mobile robot) + $(1 - \alpha)\sum$(tardiness of tasks).

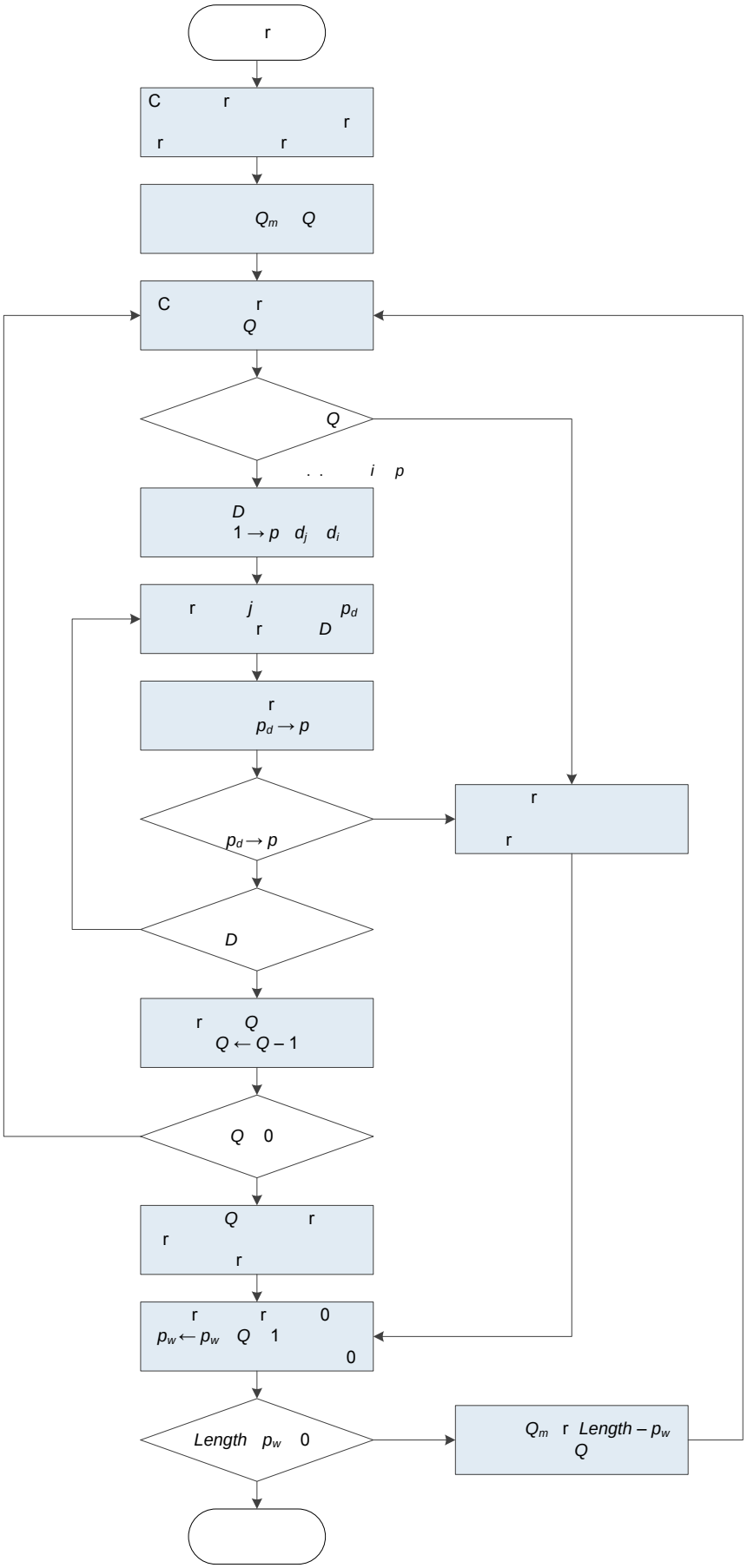

Fig. 5 Constraint handling approach

2.5.3 Genetic operators

Genetic operators mimic the process of heredity of genes to create new offspring at each generation [18]. In essence, the operators are used to alter the genetic composition of chromosomes and expected to yield improved offspring. Crossover, mutation, and selection are three main genetic operators.

Crossover operator generates offspring by combining the information contained in the parent chromosomes so that the offspring will have the desirable features from their parents. The Roulette-wheel selection is used in the algorithm, which probabilistically selects the parent chromosomes based on their fitness values. Let $F(p)$ denote the fitness value under the solution represented by parent p, then $F'(p) = \max\{F(p)|1 \leq p \leq N_p\} - F(p)$ where N_p: population size. The expected probability of parent p to be selected is given by $F'_p/\sum F'_p$.

Different crossover operators can be used on chromosomes represented by the path presentation, e.g. partially-mapped crossover (PMX), cycle crossover (CX), order crossover (OX), order-based crossover (OBX), and position-based crossover (PBX) [18]. Although the crossover operators may affect the efficiency of the search process, the quality of solutions is often reasonably close. In the presented experiment, OX operating with probability P_c is applied to generate an offspring as described below. An example illustrating the OX procedure is also shown in Fig. 6.

Step 1: Remove genes (0) representing the warehouse tasks on the parent chromosomes.

Step 2: Select two cut points on the parent chromosomes at random.

Step 3: Copy the substring between these cut points from one of the parents into the corresponding positions of the offspring.

Step 4: Fill the remaining positions of the offspring by considering the sequence of genes on the other parent starting from the second cut point. When reaching to the end of the offspring, the sequence continues at position 1. Note that during this process, a gene containing a task on the other parent will not be considered if the number of times that task is placed into the offspring is already equal to the number of requests of that task.

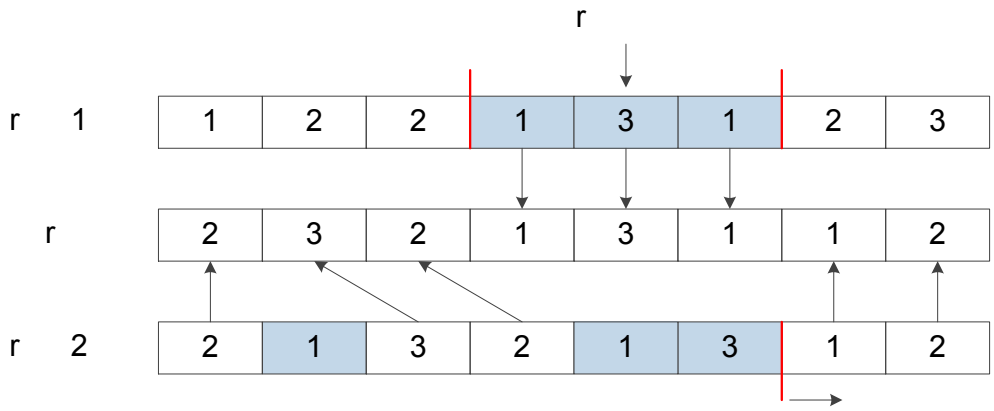

Fig. 5 Example of the OX procedure

Whenever an offspring is produced, it undergoes a mutation operator which is applied with probability P_m. The mutation selects two genes within the offspring at random and then swaps these genes to produce heterogeneous chromosomes. This procedure avoids premature convergence of the GA-based heuristic. Note that the offspring produced after crossover and mutation operations are repaired by using the constraint handling approach before they are evaluated by the fitness assignment.

For selection operator, various evolutionary methods can be applied to this problem. (μ + λ) selection is used to choose chromosomes for the next generation. Under this method, μ parents and λ offspring compete for survival, and the μ best out of the set of offspring and old parents, i.e. the μ lowest in term of the fitness value, are selected as the parents of the next generation.

2.5.4 Termination criteria

Termination criteria are used to determine when the GA-based heuristic should be stopped. On the one hand, computation time plays an important role in practice because making decision on which schedule the mobile robot should carry out tasks is a part of the real-time operations of production planners. Therefore, if the best solutions over generations do not converge, the maximum computation time CT_m would be used to stop the run. On the other hand, if the best solution does not improve over G_c consecutive generations, it would not be valuable to continue searching. The best solution up to now is then returned as the near-optimal solution.

3 Preparation

3.1 CR 1-2-3 impeller production line

The full-scale and real-world demonstration is investigated at the CR 1-2-3 impeller line that produces impellers for the CR pumps. The CR line consists of three production cells and four standard industrial step feeders where three types of parts (six vanes, one back plate, and one front plate) are assembled to an impeller. The CR line is characterized as a semi-structured environment and all communication, e.g. PLC-based signals or XML-based messages, can be transmitted via wireless network. These make a suitable demonstration site for implementing the mobile robot. Fig. 7 below depicts the CR production area where the real-world demonstration has been carried out.

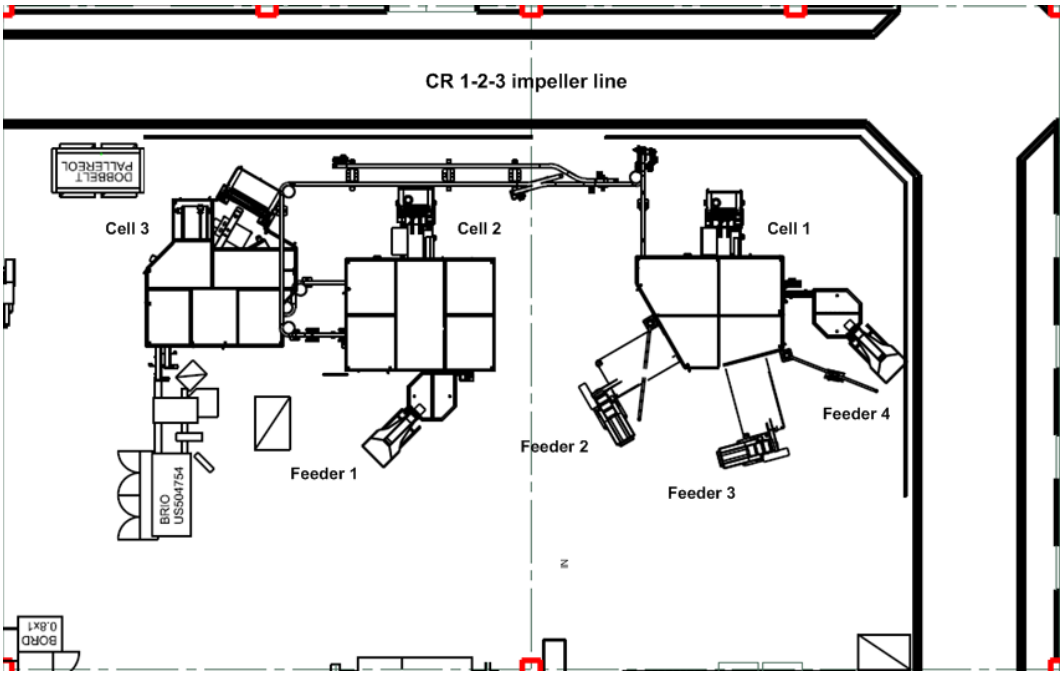

Fig. 7 CR 1-2-3 impeller production line at a pump manufacturer

3.2 Simulation and customization

Before the real-world demonstration at the CR line, 3D simulations and experiments were carried out. 3D simulation is a powerful tool for exploring what-if scenarios and for providing valuable information in advance when implementing flexible and autonomous technologies in industrial environment. 3D simulations in DELMIA Robotics were used to virtually define and test manipulation and grasping strategies, as well as virtually design and test the

mobile robot gripper, the storage of SLCs at the warehouse and transportation of SLCs on the mobile robot. Furthermore, the 3D simulations were used to verify that the mobile robot prototype is able to keep up with the production schedule at the CR line. In general, the 3D simulations proved very effective in the design phase, especially as link between academia and industry. In parallel, experiments of critical aspects, e.g. QR localization and grasping of SLCs, were conducted in different laboratory facilities [16].

Based on the requirements from the CR line and the results from the 3D simulations and laboratory experiments, it is necessary to customize the mobile robot system and the industrial environment. This is realizable based on the modular architecture of the mobile robot. The main customization steps are shown in Fig. 8.

- Customization of the mobile robot: a) *tooling*: design and use a robot gripper compliant with the utilized SLC; b) *SLC rack*: enable the mobile robot to carry and transport several SLCs at a time; c) *vision*: use a mounted camera to acquire images containing QR codes on SLCs and implement standard algorithms for QR identification and localization.
- Customization of the industrial environment: a) *safety*: use fences and warning signs to ensure that no one enters the demonstration area as well as prevent the mobile robot leaving that area; b) *navigation and localization*: place reflector landmarks at the workstations; c) *warehouse*: customize the SLC storage consisting of two tables with four slots on each table.

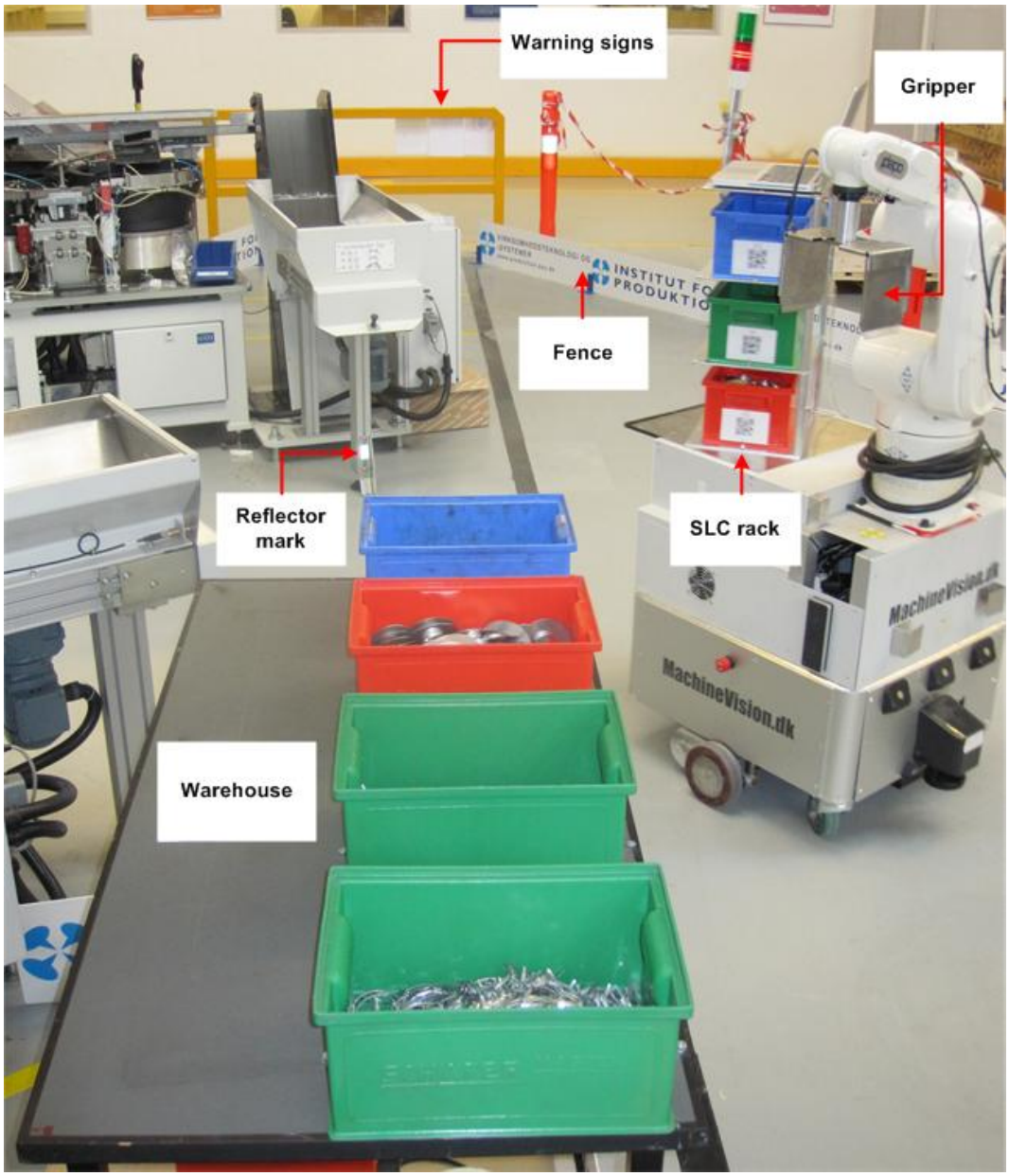

Fig. 8 Customization for the real-world demonstration

3.3 Mission planner and controller

Another important part of the real-world demonstration is the interaction and communication between the mobile robots and the manufacturing system or shop-floor operators. By integrating the mobile robot into the general manufacturing network, it is possible to plan and control globally, as the mobile robot becomes a resource on the same level as corresponding manufacturing device. A program named Mission Planner and Controller (MPC) was developed to handle this integration.

The MPC consists of two main modules: Mission Planner (MP) and Mission Controller (MC). The MP module, into which the GA-based heuristic

(the decision-making nucleus) is integrated, is responsible for quickly generating best schedules under considerations of practical constraints. The MC module has responsibility for transforming generated schedules into real schedules/executions based on the feedback from the mobile robot. First, the input data is derived from a shop-floor operator or from Manufacturing Execution System (MES). Then, the MP module uses the input data to generate the best schedule which serves as an input to the MC module (this best schedule is also shown to the operator). Next, the MC module uses XML-based TCP/IP communication via a wireless network to command the mobile robot to execute tasks in succession according to the best schedule and feedback from the mobile robot. In practice, there might be some unexpected events such as errors in manufacturing cell (e.g. machine breakdown) or changes in manufacturing condition (e.g. cycle time of production lines). These events will be reported by the MES so that the MC module is able to update current status of the shop floor and then call the MP module to reschedule feeding tasks of the mobile robot. The MP module will in turn use the current status as new input, re-optimize to find an alternative schedule, and send this schedule back to the MC module for real executions [5]. Fig. 9 below depicts the overall structure of the MPC and its communication with the MES and mobile robot.

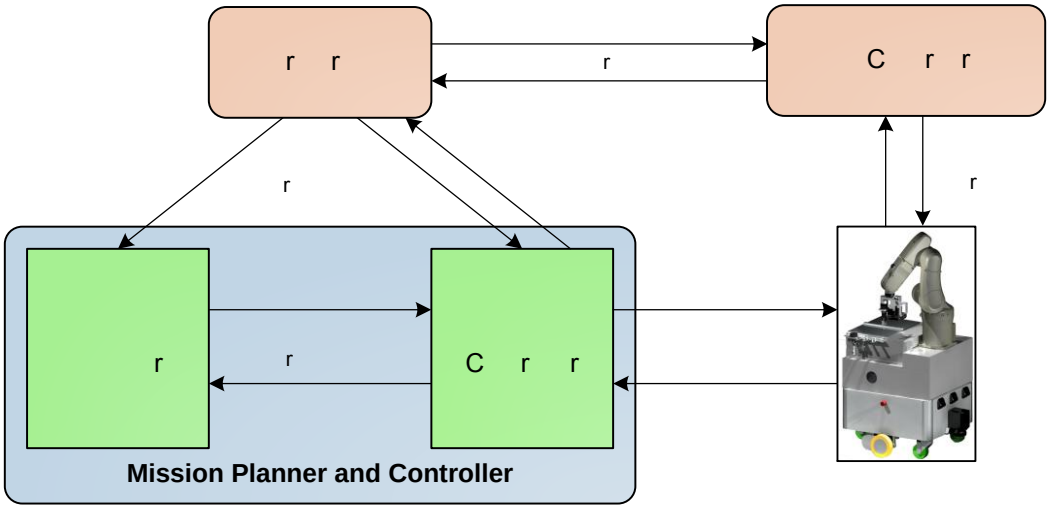

Fig. 9 The overall structure and communication of the MPC

4 Numerical Experiments

To examine the methodology, specifically the performance of the MIP model and GA-based heuristic, a real-world demonstration and computational experiments are conducted in this section. The real-world demonstration including two

different cases has been investigated with real data of Grundfos A/S, a Danish company which is one of the world's leading pump manufacturers. The extension of the demonstration considering several reasonable assumptions has been also conducted to make the evaluation of the proposed approaches more convincing. Finally, various problem instances are randomly generated and tested in order to provide more persuasive evidence of the performance of the proposed approaches. In the experiments, the MIP model has been coded and solved by the mathematical modelling language ILOG CPLEX, while the GA-based heuristic has been programmed in VB.NET. All the experiments run on a PC having an Intel® Core i5 2.67 GHz processor and 4 GB RAM.

4.1 Real-world demonstration

The methodology was tested for a period of three days at the CR 1-2-3 impeller production line. The following data are taken from the MES as well as real tests on the shop floor of the CR factory. This data is used as input for two cases of the demonstration. The average number of parts per SLC fed to feeder 1 or 4 is 125 (approximately 2 kg/SLC), while the average number of parts per SLC fed to feeder 2 and 3 is 1100 (approximately 1 kg/SLC). The maximum levels, minimum levels, and part-feeding rates to machines of feeders are given in Table 1. The part-feeding rates to machines of feeders are derived from the cycle time of the CR line of 4.5 seconds [16]. Specifically feeders 1 and 4 feed machines with one back plate and one front plate every 4.5 seconds, while feeders 2 and 3 feed machines with one vane every 1.5 seconds (3 vanes for every 4.5 seconds). The working times of the mobile robot at the feeders are given in Table 2, and Table 3 shows the travelling times of the mobile robot from locations to locations (feeder 0 means the warehouse).

Table 1 Maximum levels, minimum levels and part-feeding rates to machines of feeders

Feeder/Task	1	2	3	4
Maximum level (part)	250	2000	2000	250
Minimum level (part)	125	900	900	125
Part-feeding rate (second/part)	4.5	1.5	1.5	4.5

Table 2 Working times of robot at locations (seconds)

Feeder/Task	0	1	2	3	4
Working time of robot	110	42	42	42	42

Table 3 Travelling times of robot from locations to others (seconds)

From feeder \ To feeder	0	1	2	3	4
0	0	50	58	45	34
1	49	0	56	48	59
2	58	56	0	32	45
3	42	49	32	0	42
4	34	60	44	42	0

In the initial design, the mobile robot has the capability to carry up to three SLCs at a time while performing the multiple-part feeding tasks at the feeders. Hence, two different cases of the demonstration have been investigated corresponding to two maximum numbers of SLCs, Q_m = 2 and Q_m = 3, and with the planning horizon T of approximately 35 minutes due to the robot's battery limit. In each case of the demonstration, the total number of requests of the multiple-part feeding tasks is 8, and the number of decision variables is 2112. Furthermore, two situations of the MIP are investigated for comparing the performances with the GA-based heuristic. The first situation is carried out when giving the same maximum computation time CT_m as the GA-based heuristic while the second situation is performed without limit on the computation/run time. The weighted coefficient α is set to be 0.2 because the first objective of minimizing the total travelling time of the robot is less important than the second objective of minimizing the total tardiness of the tasks. For GA parameters, pilot runs have been utilized to decide on the values of N_p, P_c, P_m, G_c, and CT_m. Each parameter is tested at four different levels. These are respectively: N_p (50, 100, 150, 200), P_c (0.4, 0.6, 0.8, 1.0), P_m (0.05, 0.1, 0.15, 0.2), G_c (50, 100, 150, 200), and CT_m in seconds (15, 30, 45, 60). There are ten observations under each level, and the 40 runs of each parameter are made in random. The best performance of the GA-based heuristic in terms of the weighted objective value and computation time is obtained at 100, 0.8, 0.1, 100, and 60 for N_p, P_c, P_m, G_c, and CT_m, respectively. Table 4 gives solutions of the MIP and GA-based heuristic on the weighted objective value (in seconds) and computation time (in seconds) for the demonstration at the CR line. Sequences of the multiple-part feeding tasks found by the heuristic are depicted using Gantt charts in Fig. 10.

Table 4 Solutions of the demonstration under MIP and GA-based heuristic

Case	Q_m	MIP (limited to CT_m)				MIP (unlimited)				GA-based heuristic				Penalty of heuristic (%) vs. MIP (un-limited)
		Travelling time	Tardiness	Weighted objective value	Com. Time	Travelling time	Tardiness	Weighted objective value	Com. Time	Travelling time	Tardiness	Weighted objective value	Com. Time	
1	2	814.00	723.88	741.90	60.00	534.00	0.00	106.80	13975.18	561.00	0.00	112.20	<1	5.06
2	3	739.00	0.00	147.80	60.00	399.00	0.00	79.80	4598.16	418.00	0.00	83.60	<1	4.76

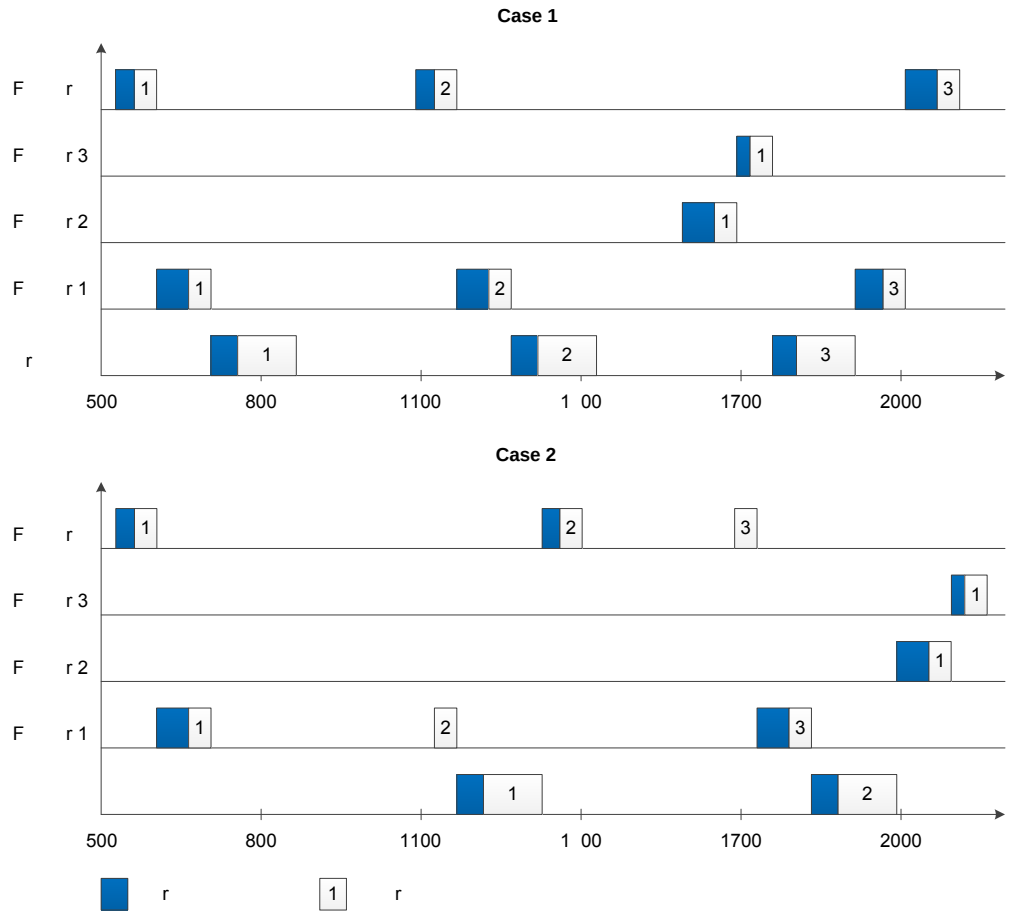

Fig. 10 Gantt charts for the solutions of two cases of the demonstration

From Table 4, it can be seen that when giving the same maximum computation time CT_m as the GA-based heuristic, the MIP found feasible solutions for the two cases of the demonstration. However, the solutions found by the MIP are much worse than those found by the proposed heuristic. Even in case 1 the solution found by the MIP incurs a tardiness of 723.9 s/12.1 min while the solutions found by the heuristic do not incur any tardiness of tasks. The weighted objective values found through the heuristic are greater than those found by the MIP when the run time of the MIP is unlimited. However, the differences are only about 5%, and this is deemed to be an acceptable error. Furthermore, the computation time shows that use of the MIP is too time-consuming while the heuristic significantly faster obtains near-optimal solutions (approximately 4 hours in case 1 or 1.3 hours in case 2 as opposed to less than a second). It also reveals that the higher maximum numbers of SLCs the robot can carry, the less it has to travel around the production cell (561 s/9.4 min with Q_m of 2 as opposed to 418 s/7 min with Q_m of 3).

Practically, the demonstration on the shop floor of the CR factory was a success. The mobile robot prototype was able to continuously pick and place SLCs from/to the warehouse and empty them into different feeders of the CR line. This sequence of serving feeders was practically based on the near-optimal schedules (with no tardiness) generated from the GA-based heuristic in the MP module and then sent through the MC module of the MPC. The main performance characteristics of the mobile robot in the demonstration are listed as follows: a) mobile platform accuracy: ± 10 mm; b) effective platform payload: 50 kg; c) manipulation accuracy (vision guided): ± 1 mm; d) effective manipulator payload: 2 kg.

4.2 Computational experiments

In this section, the performance of the MIP model and GA-based heuristic will be tested on a large number of problem instances. Firstly, the demonstration is extended by assuming that the mobile robot has capability of carrying up to 4 SLCs at a time, and the battery limit of the mobile robot allows it to work up to 8 hours (a full production shift). Further comparisons of the weighted objective value and computation time for the MIP and GA-based heuristic are presented in Table 5. Note that in this extended demonstration, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs. The cells containing a "–" symbol indicate that the results of the corresponding problems cannot be obtained by using the corresponding method. The total number of requests of all tasks and number of decision variables are also given. From Table 5, it can be observed that the proposed heuristic has the capability of solving larger problems while the MIP cannot find any feasible solution for problems of this scale. It also shows that in case of full production shift of 8 hours, the proposed heuristic is able to find the best solutions in less than 40 seconds. Furthermore, the standard deviation of the weighted objective value is quite small in comparison with the average. The GA-based heuristic, therefore, demonstrates efficiency in solving the larger problems.

Table 5 Comparison between MIP and GA-based heuristic for extension of the demonstration

Q_m	T	Total requests of tasks	Total variables	MIP				GA-based heuristic					
				Travelling time	Tardiness	Weighted objective value	Com. Time	Travelling time	Tardiness	Weighted objective value		Com. Time	
										Average	SD	Average	SD
2	2	32	131328	-	-	-	-	2049.03	0.00	409.81	0.04	2.70	0.48
	4	66	1150512	-	-	-	-	4105.57	0.00	821.11	9.15	17.58	3.95
	8	136	10062912	-	-	-	-	8864.17	0.00	1772.83	9.79	36.32	6.21
3	2	32	131328	-	-	-	-	1555.30	0.00	311.06	0.92	4.93	1.50
	4	66	1150512	-	-	-	-	3448.30	0.00	689.66	13.96	13.53	4.92
	8	136	10062912	-	-	-	-	7669.80	0.00	1533.96	10.70	36.62	6.65
4	2	32	131328	-	-	-	-	1473.00	0.00	294.60	1.43	3.69	0.77
	4	66	1150512	-	-	-	-	3361.30	0.00	672.26	13.50	12.37	4.67
	8	136	10062912	-	-	-	-	7227.67	0.00	1445.53	17.83	32.64	8.05

Secondly, 10 problems are generated with different numbers of feeders, maximum numbers of SLCs, planning horizons, and other system parameters. The number of feeders and the maximum number of SLCs are randomly generated in the ranges of [3, 10] and [2, 4], respectively. The planning horizons in hours are 2, 4, and 8 corresponding to a quarter, half, and full of the production shift. The maximum and minimum levels of parts in feeders are respectively uniformly distributed within the ranges of [500, 2000] and [100, 1000] while part-feeding rates to machines of feeders (in seconds) are generated in the interval [1.5, 6.5]. The working times of the robot in seconds at feeders and the warehouse are respectively distributed within the range of [40, 60] and [100, 120] while the travelling times of the robot in seconds are generated in the interval [30, 70]. Note that the time/cost matrices of the generated travelling times should satisfy the triangle inequality. The comparisons between the MIP and GA-based heuristic for 10 randomly generated problems are presented in Table 6. Similar to the extension of the demonstration, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs. General information for these 10 problems is also shown in Table 6.

It can be seen from Table 6 that the GA-based heuristic is superior to the MIP for large problems. The MIP found feasible solutions for problem instances 1 and 3. However, the solutions found by the MIP are much worse than those found by the proposed heuristic. Furthermore in problem instances 1 and 3, the solutions found by the MIP incurs tardiness while those found the proposed heuristic do not incur any tardiness. For the other problems, the MIP cannot find any feasible solution. The GA-based heuristic, by contrast, is able find best solutions for all 10 problem instances. These best solutions do not incur tardiness except problem instance 10. Moreover, in terms of the objective value, the standard deviation is quite small in comparison with the average. These results provide more persuasive evidence to prove that the GA-based heuristic performs effectively.

Table 6 Comparison between MIP and GA-based heuristic for 10 randomly generated problems

No.	Number of feeders	Q_m	T	Total requests of tasks	Total variables	MIP				GA-based heuristic					
						Travelling time	Tardiness	Weighted objective value	Com. Time	Travelling time	Tardiness	Weighted objective value		Com. Time	
												Average	SD	Average	SD
1	3	2	2	7	1428	962.00	615.15	684.52	60.00	477.17	0.00	95.43	0.18	0.28	0.03
2	8	4	2	21	37212	-	-	-	-	1090.07	0.00	218.01	2.26	1.59	0.27
3	4	3	2	10	4080	971.00	1565.20	1446.36	60.00	344.17	0.00	68.83	2.37	0.36	0.05
4	6	2	4	54	630288	-	-	-	-	2539.03	0.00	507.81	8.90	10.97	3.12
5	4	4	4	32	131328	-	-	-	-	1211.93	0.00	242.39	5.57	4.12	0.91
6	9	3	4	75	1688100	-	-	-	-	3690.90	0.00	738.18	11.04	26.98	7.60
7	7	4	8	109	5180988	-	-	-	-	4990.77	0.00	998.15	10.96	34.51	8.17
8	8	3	8	128	8389632	-	-	-	-	7589.47	0.00	1517.89	13.83	60.00	0.00
9	5	2	8	96	3539712	-	-	-	-	6311.43	0.00	1262.29	11.84	24.09	5.50
10	10	3	8	150	13501200	-	-	-	-	9797.30	1.32	1960.52	13.87	60.00	0.00

5 Conclusions

The paper studies the problem of implementation of an autonomous industrial mobile robot in a real-world application. The multi-criteria optimization problem of scheduling tasks of the mobile robot is also considered. The main novelties of this research lie in the transfer of the mobile robot technology from laboratory experiments to real-world applications and in the scheduling problem with the simultaneous consideration of soft time windows of tasks and limit on carrying capacity of the mobile robot. A methodology was proposed to assign the mobile robot to the production environment. This methodology consists of a mobile robot system design (Little Helper prototype), a suitable industrial application (multiple-part feeding), an implementation concept for industrial environment (the Bartender Concept), an MIP model, and a GA-based heuristic. The MIP model is used to find exact optimal solutions for the problem of scheduling multiple-part feeding tasks of the mobile robot. Due to the NP-hard nature of the scheduling problem, this mathematical method is only applicable to small-scale problems with few feeders and a short planning horizon. A genetic algorithm-based heuristic was then proposed to find near-optimal solutions. The quality of these solutions could then be evaluated by using the MIP solutions as reference points to quantify the scale of benefits. The real-world demonstration at an impeller production line and more computational experiments were described to demonstrate the effectiveness of the methodology, specifically the MIP model and GA-based heuristic. Overall, the real-world demonstration was a success as the experiments showed that the mobile robot was capable of continuously performing meaningful industrial tasks. The results of the demonstration also showed that use of the MIP was too time-consuming while the proposed heuristic was significantly faster in obtaining near-optimal solutions. Further experiments provided persuasive evidence that the proposed heuristic is capable of solving problems of various sizes and more efficient than the MIP in terms of the weighted objective value when giving the same maximum computation time. These solutions are useful for decision making at operational levels, and the proposed heuristic could be also applied in a variety of tasks of not only mobile robots but also automatic guided vehicles or unmanned aerial vehicles. For further research, the mobile robot will be improved to perform more advanced tasks such as pre-assembly, quality inspection, and machine tending. Consequently, a general

model of scheduling a fleet of mobile robots to perform these advanced tasks should be taken into account.

Acknowledgements This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

Reference

[1] N. Ascheuer, L.F. Escudero, M. Grötschel, M.A. Stoer, Cutting plane approach to the sequential ordering problem (with application to job scheduling in manufacturing), SIAM Journal of Optimization. 3 (1993) 25–42.

[2] G. Carpaneto, M. Dell'Amico, P. Toth, Exact solution of large-scale, asymmetric traveling salesman problems, ACM Transactions on Mathematical Software. 21 (1995) 394–409.

[3] I.C. Choi, S.I. Kim, H.S. Kim, A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem, Computers and Operations Research. 30 (2003) 773–786.

[4] Q.V. Dang, I. Nielsen, K. Steger-Jensen, Mathematical formulation for mobile robot scheduling problem in a manufacturing cell, in: J. Frick, B.T. Laugen (Eds.), Advances in Production Management Systems. Value Networks: Innovation, Technologies, and Management, IFIP AICT, vol. 384, Springer, Heidelberg, 2012, pp. 37–44.

[5] Q.V. Dang, I. Nielsen, Simultaneous scheduling of machines and mobile robots, in: J.M. Corchado JM et al (Eds), PAAMS 2013 Workshops, CCIS, vol. 365, Springer, Heidelberg, 2013, pp. 118–128.

[6] Q.V. Dang, I. Nielsen, K. Steger-Jensen, O. Madsen, Scheduling a single mobile robot for part-feeding tasks of production lines, Journal of Intelligent Manufacturing. (2013) doi:10.1007/s10845-013-0729-y.

[7] S. Datta, R. Ray, D. Banerji, Development of autonomous mobile robot with manipulator for manufacturing environment, International Journal of Advanced Manufacturing Technology. 38 (2008) 536-42.

[8] Y. Edan, T. Flash, U.M. Peiper, I. Shmullevich, Y. Sarig, Near-minimum-time task planning for fruit-picking robots, IEEE Transactions on Robotics and Automation. 7 (1991) 48–55.

[9] X. Geng, Z. Chen, W. Yang, D. Shi, K. Zhao, Solving the travelling salesman problem based on an adaptive simulated annealing algorithm with greedy search, Applied Soft Computing. 11 (2011) 3680–3689.

[10] R. Germs, B. Goldengorin, M. Turkensteen, Lower tolerance-based branch and bound algorithms for the ATSP. Computer and Operations Research. 39 (2012) 291–298.

[11] D.E. Goldberg, Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, New York, 1989.

[12] A. Hentout, B. Bouzouia, I. Akli, R. Toumi, Mobile manipulation: a case study, in: A. Lazinica, K. Hiroyuki (Eds.), Robot Manipulators, New Achievements, INTECH, Rijeka, 2010, pp. 145–167.

[13] High Level Group, Manufuture: Strategic Research Agenda. High Level Group, Luxembourg, 2006.

[14] J. Hurink, S. Knust, A tabu search algorithm for scheduling a single robot in a job-shop environment, Discrete Applied Mathematics. 119 (2002) 181–203.

[15] M. Hvilshøj, S. Bøgh, 'Little Helper' – an autonomous industrial mobile manipulator concept International Journal of Advanced Robotic Systems. (2011) doi: 10.5772/10579.

[16] M. Hvilshøj, S. Bøgh, O.S. Nielsen, M. Madsen, Multiple part feeding – real-world application for mobile manipulators, Assembly Automation. 32 (2012) 62–71.

[17] D. Katz, J. Kenney, O. Brock, How can robots succeed in unstructured environments?, in: Workshop on Robot Manipulation: Intelligence in Human Environments at Robotics: Science and Systems, Zurich, Switzerland, 2008.

[18] L. Lin, S.W. Shinn, M. Gen, H. Hwang, Network model and effective evolutionary approach for AGV dispatching in manufacturing system, Journal of Intelligent Manufacturing. 17 (2006) 465–477.

[19] F. Liu, G. Zeng, Study of genetic algorithm with reinforcement learning to solve the TSP, Expert Systems with Applications. 36 (2009) 6995–7001.

[20] O. Maimon, D. Braha, V. Seth, A neural network approach for a robot task sequencing problem, Artificial Intelligence in Engineering. 14 (2000) 175–189.

[21] S. Mekid, T. Schlegel, N. Aspragathos, R. Teti, Foresight formulation in innovative production, automation and control systems, Foresight. 9 (2007) 35-47.

[22] C. Moon, J. Kim, G. Choi, Y. Seo, An efficient genetic algorithm for the traveling salesman problem with precedence constraints, European Journal of Operational Research. 140 (2002) 606–617.

[23] J.W. Ohlmann, B.W. Thomas, A compressed-annealing heuristic for the traveling salesman problem with time windows, INFORMS Journal on Computing. 19 (2007) 80–90.

[24] J. Schuler, Integration von Förder- und Handhabungseinrichtungen, Springer, Berlin, 1987.

[25] E.A. Silver, D.F. Pyke, R. Peterson, Inventory management and production planning and scheduling, John Wiley and Sons, New York, 1998.

[26] L.V. Snyder, M.S. Daskin, A random-key genetic algorithm for the generalized traveling salesman problem, European Journal of Operational Research. 174 (2006) 38–53.

[27] A. Stopp, S. Horstmann, S. Kristensen, F. Lohnert, Towards interactive learning for manufacturing assistants, IEEE Transactions on Industrial Electronics, 50 (2003) 705–707.

[28] M. Turkensteen, D. Ghosh, B. Goldengorin, G. Sierksma, Tolerance-based branch and bound algorithms for the ATSP, European Journal of Operational Research. 189 (2008) 775–788.

[29] L.N. Xing, Y.W. Chen, K.W. Yang, F. Hou, X.S. Shen, H.P. Cai, A hybrid approach combining an improved genetic algorithm and optimization strategies for the asymmetric traveling salesman problem, Engineering Applications of Artificial Intelligence. 21 (2008) 1370–1380.

[30] P.Th. Zacharia, N.A. Aspragathos, Optimal robot task scheduling based on genetic algorithms, Robotics and Computer-Integrated Manufacturing. 21 (2005) 67–79.

Paper J

Simultaneous scheduling of machines and mobile robots

This paper has been published in PAAMS 2013 Workshops, CCIS 365. Springer-Verlag Berlin Heidelberg, pp. 118–128

Simultaneous Scheduling of Machines and Mobile Robots

Quang-Vinh Dang and Izabela Nielsen

Dept. of Mechanical and Manufacturing Engineering, Aalborg University, Aalborg, Denmark
{vinhise,izabela}@m-tech.aau.dk

Abstract. This paper deals with the problem of simultaneously scheduling machines and a number of autonomous mobile robots in a flexible manufacturing system (FMS). Besides capability of transporting materials between machines, the considered mobile robots are different from other material handling devices in terms of their advanced ability to perform tasks at machines by using their manipulation arms. The mobile robots thus have to be scheduled in relation to scheduling of machines so as to increase the efficiency of the overall system. The performance criterion is to minimize time required to complete all tasks or makespan. A heuristic based on genetic algorithm is developed to find the best solution for the problem. A numerical example is investigated to demonstrate results of the proposed approach. The implementation of the proposed approach in a multi-agent system is also generally described.

Keywords: Scheduling, Mobile Robots, Genetic Algorithm, FMS.

1 Introduction

The automation technology in combination with advances in production management has dramatically changed the equipment used by manufacturing companies as well as the issues in planning and control. With these changes, highly automated and unmanned production systems have become more popular in several industrial areas, e.g., automotive, chemical, robot, and pump manufacturing. An automatic production system consists of intelligent and flexible machines and mobile robots grouped into cells in such a way that entire production of each product can be performed within one of the cells. Mobile robots capable of moving around within their environment and not fixed to one physical location are among various advanced material handling techniques that are finding increasing applications in today manufacturing. However, besides transporting a variety of part types from one point to another without human intervention, mobile robots are more significantly advanced than automated guided vehicles (AGVs) in the capability of performing various value-added tasks on different machines (or workstations) based on their manipulation arms. The tasks include such processes as machine tending, pre-assembly, and quality inspection. Moreover, using mobile robots can lead to production efficiency gains such as less energy usage or lower tool-changing costs than typical industrial robots. The advanced abilities of the mobile robots pay the way for establishing transformable production systems that combine the best features of both fully automated and strictly manual manufacturing

J.M. Corchado et al. (Eds.): PAAMS 2013 Workshops, CCIS 365, pp. 118–128, 2013.

environments. In this paper, a particular problem is taken into account. The problem consists of a number of operations of different tasks which are processed on flexible machines. The processes of the operations need the participation of the mobile robots which not only transport parts of tasks from one machine to another where needed but also perform the operations of tasks on the machines. In that context, mobile robots play the role of agents, attempting to reach goals while following rules specific for a given production system. The considered systems are thus treated as multi-agents ones in which each robot can be seen as an autonomous object capable of undertaking decisions about moving, feeding, and completing operations, etc. [6]. However, to utilize the systems in an efficient manner requires the ability to properly schedule operations of tasks on machines and agents. Hence, it is important to plan in which sequence the machines and agents process the operations so that performance criteria can be achieved while satisfying a number of practical constraints.

The problem of simultaneous scheduling of machines and mobile robots has been modeled in several respects comparable to the problems of simultaneous scheduling of machines and AGVs which have attracted interest of researchers in recent decades. Ulusoy and Bilge [13] and Bilge and Ulusoy [2] propose an iterative heuristic based on the decomposition of the master problem into two sub-problems. Ulusoy et al. [14] describe a genetic algorithm (GA) approach for concurrent scheduling machines and AGVs. Abdelmaguid et al. [1] introduce a hybrid method composed of a GA for scheduling of machines and a heuristic for scheduling of vehicles. Jerald et al. [10] deal with the problem of scheduling of parts and AGVs in an FMS environment using adaptive GA. Reddy and Rao [12] present a hybrid multi-objective GA for scheduling of machines and AGVs in FMS. Lin et al. [11] model an AGV system by using network structure and propose an effective evolutionary approach for solving a kind of AGV problems. Deroussi et al. [8] develop a simple metaheuristic approach in which a new solution representation based on vehicles rather than machines to solve the problem of simultaneous scheduling of machines and AGVs. Bocewicz et al. [3-4] deal with the problem of AGV operation synchronization mechanism in FMSs where transport processes can be modeled as a system of cyclic concurrent processes sharing common resources, e.g., machines. Bocewicz and Banaszak [5] present a new modeling framework enabling to prototype and evaluate multimodal cyclic processes, e.g., vehicles, which share common machines while comparing their cyclic steady state.

Although much related research has been completed, the problem of simultaneous scheduling of machines and mobile robots has not yet been studied in the literature. Furthermore, the surveyed genetic algorithms to agent-based solutions are not well suited and cannot be directly used to solve this problem due to the lack of a suitable mechanism to simultaneously assign tasks to machines and mobile robots while taking into account precedence and routing constraints. In this problem, the considered mobile robots have not only the capability of transporting parts of tasks similar to other material-handling devices but also the advanced capability of performing tasks at destination machines by using their manipulation arms. In other words, after transporting parts of the tasks to the destination machines, the mobile robots have to stay at those machines and complete performing the tasks before moving to other places. Moreover, this problem is composed of two interrelated decision problems that are scheduling of machines and scheduling of mobile robots (or vehicles). Both problems

are known to be NP-hard [8] resulting in a more complicated NP-hard problem when they are taken into account simultaneously. In that context, our main contribution is to develop a computationally efficient heuristic based on genetic algorithm, a promising algorithm to this class of problems, so as to find the best solutions for the problem of simultaneous scheduling machines and mobile robots. Compared to other optimization methods, the major benefit of GA regards multiple directional search using a set/population of candidate solutions which enables GA to search in several directions simultaneously. In this way, many paths to the optimum are processed in parallel that leads to a clear improvement in performance. Furthermore, since information from many different regions is used, GA is resistant to remain trapped in a suboptimal solution and able to move away from it if the population finds better solutions in other areas. In this paper, the best solutions achieved by the genetic algorithm-based heuristic are useful for decision making at operational levels and the proposed approach enables to model and evaluate scheduling tasks of multi-agent systems.

The remainder of this paper is organized as follows: in the next section, problem description is presented while a heuristic based on genetic algorithm is developed in Section 3. A numerical example is conducted to demonstrate results of the proposed approach in Section 4. Section 5 generally describes how the proposed approach has been implemented and interacted with other components in a multi-agent system. Finally, conclusions and future research directions are drawn in Section 6.

2 Problem Description

The work is developed for an FMS which products parts or components for the pump manufacturing industry at a factory. In the FMS, a set of independent tasks has to be processed without pre-emption on a set of machine tools along with a set of identical mobile robots (agents). Each task consists of a sequence of operations. Each machine and each mobile robot can process only one operation at one time, and each operation can be processed by only one machine and one mobile robot at the same time. Note that in this FMS processing operations at machines has the participation of mobile robots. This results from the mobile robots' advanced abilities which enable them not only to transport parts from one machine to another but also performing operations on the machines to which the parts are transported. The considered FMS can be seen as a multi-agent system.

To enable the construction of a simultaneous schedule for the machines and mobile robots, assumptions are considered as follows:

- Each task is available at the beginning of the scheduling period.
- Each operation sequence of each task (the routing of each part type) is available before making scheduling decisions.
- Each mobile robot can transport only one kind of parts at a time.
- There is sufficient output buffer space at each machine.
- Traveling time is only machine-dependent and deterministic. Processing time is also deterministic.
- Such issues as traffic congestions, mobile robot collisions, machine failures, scraps are not considered in this paper.

Making decisions on scheduling machines and mobile robots simultaneously is a part of the real-time activities of production planners. It means that the best solution must be quickly obtained at the beginning of (re)scheduling periods. Furthermore, concerning the problem belonging to NP-hard class, computation time exponentially grows with the size of the problem, e.g., more number of tasks, machines, or mobile robots. It is therefore necessary to develop a computationally effective algorithm, namely a GA-based heuristic, which determines in which sequence the machines and mobile robots should handle tasks so as to minimize time required to complete all tasks or makespan while satisfying a number of practical constraints.

3 Genetic Algorithm-Based Heuristic

In this section a genetic algorithm, a broadly applicable search approach imitating the evolutionary process in nature, is used to develop a heuristic which is allowed to convert the mentioned problem to the way that best solutions could be found. In GAs, each individual solution is represented in the form of a finite length string called a chromosome. A chromosome is composed of a set of locations known as genes that contain discrete values pertaining to a problem solution. Through the use of genetic operators such as crossover, mutation, and selection to the chromosomes of selected solutions are in a systematic fashion to generate a new generation of solutions moving towards the optimization of certain criteria [9]. The GA-based heuristic shown in Fig. 1 consists of the following main steps: genetic representation; initialization; decoding operator and fitness evaluation; genetic operators including selection, crossover, and mutation; repair operator; termination criteria.

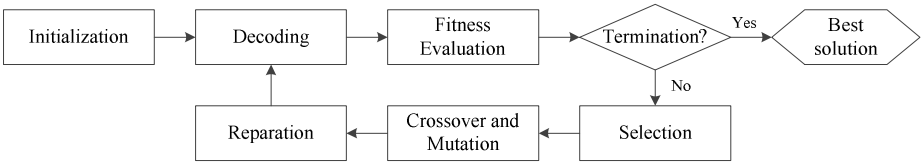

Fig. 1. Flow chart of GA-based heuristic

3.1 Genetic Representation

One of the main concerns when applying GAs to an optimization problem is to find an appropriate coding scheme that transforms feasible solutions into representations amenable to genetic search and reversibly decode these representations [1]. For the problem under consideration, a feasible solution can be encoded by a chromosome representing both operation sequencing and mobile robot assignment. Consequently, each gene in the chromosome is made up of two parts. The first part refers to an operation on a specific machine which is assumed to be scheduled at its earliest starting time. The second part identifies the mobile robot that will transport parts from any machine to that machine and then perform that operation on that machine. The chromosome length is equal to the total number of operations of all tasks. Fig. 2 below illustrates a feasible chromosome of an example problem with 5 operations of 2 tasks, 3 machines, and 2 mobile robots.

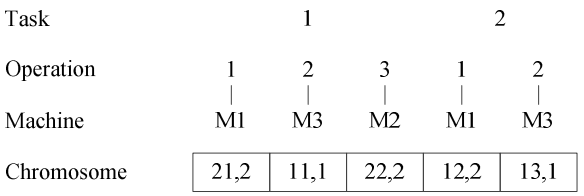

Fig. 2. Illustration of a feasible chromosome

3.2 Initialization

Random chromosomes are generated for providing solutions to the initial population. A chromosome is constructed of gene by gene. Each gene is first assigned an eligible operation (an operation is said to be eligible if all its predecessors are assigned). Then, one of the mobile robots is randomly chosen to complete the gene. If the chromosome is not yet complete, the eligible set of operations is updated and the process continues. The pseudocode of the initialization method is given below.

```
Procedure: Initialization
Begin
  D ← {op_i | op_i: first operations of all tasks};
  R ← {1,...,n_r};//n_r: number of mobile robots;
  Repeat
    Select an operation op_i ∈ D;
    Assign op_i to a mobile robot mr ∈ R;
    D ← D/{op_i};
    If (successor of op_i exists) Then
      D ← D ∪ {successor of op_i};
  Until (D ≔ ∅)
End
```

3.3 Decoding Operator and Fitness Evaluation

The decoding operator and fitness evaluation are described in the pseudocode below.

```
Procedure: Decoding and Fitness Evaluation
Begin
  For i = 1 To chromosome_length
    op ← operation at location i in the chromosome;
    mc ← machine of op;
    pd ← predecessor of op in the task sequence;
    mr ← mobile robot of op;
    If (pd ≔ ∅) Then st ← 0;
    Else st ← pd.completion_time;
    ns ← number of scheduled operations on machine mc;
    If (ns > 0) Then
      ls ← last operation scheduled on machine mc;
      ct ← ls.completion_time;
```

```
      If (st < ct) Then st ← ct;
    sr ← mr.last_work_destination;
    If (pd ≠ ∅) Then md ← machine of pd;
    Else md ← or;
    et ← travel time between sr and md;
    lt ← travel time between md and mc;
    ft ← Max(pd.completion_time, mr.last_work_time) + et
                                                    + lt;
    If (st < ft) Then st ← ft;
    op.completion_time ← st + op.processing_time;
    mr.last_work_time ← op.completion_time;
    Schedule op on machine mc and mobile robot mr;
  End
  C_max ← Max(op_j.completion_time | op_j: last operations of
                                              all tasks);
End
```

3.4 Genetic Operators

Selection, crossover, and mutation are three main genetic operators. For selection, various evolutionary methods can be applied to this problem. $(\mu + \lambda)$ selection is used to choose chromosomes for reproduction. Under this method, μ parents and λ offspring compete for survival and the μ best out of the set of offspring and old parents, i.e. the μ lowest in term of the makespan, are selected as the parents of the next generation. This selection method guarantees that the best solutions up to now are always in the parent generation [6-7].

Crossover operator generates offspring by combining the information contained in the parent chromosomes so that the offspring will have desirable features from their parents. The Roulette-wheel selection is first used to select the parent chromosomes based on their fitness values. Then, a uniform crossover operated with probability P_c will be used to generate offspring as follows. Starting from the first operations on the parents, iteratively, one of the parents is randomly selected. The next unconsidered operation of the selected parent becomes the next operation on the first offspring while the next unconsidered operation of the other parent is the next operation of the second offspring. If the mobile robot selected for that operation is the same on both parents, then that selection is also made on the child; if not, one of the mobile robots of the parents is randomly chosen. Fig. 3 below depicts the uniform crossover.

Mutation operator produces spontaneous random changes in various chromosomes. For the current encoding method, there are two mutation operators, one for each part of a gene and with a probability P_m. The first mutation operator selects two random positions on a chromosome and swaps the operations with respect to those positions. Note that the chromosome may be infeasible in terms of precedence constraints after the operation mutation. Hence it has to be adjusted by using the repair operator in Section 3.5. The second mutation operator replaces the mobile robot assignment at a gene with one of the mobile robots which is randomly chosen. This may lead to the same mobile robot assignment for a particular gene, and aim to prevent the loss of any good assignment.

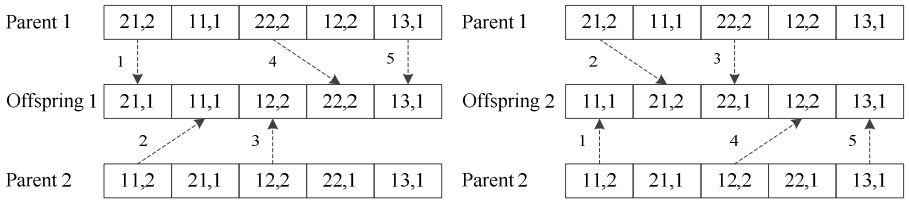

Fig. 3. Uniform crossover

3.5 Repair Operator

A repair operator is developed to validate chromosomes with any precedence violations after the mutation operators. This operator involves the exchange of locations of operations belonging to the same task such that a valid sequence of operations is achieved. The following pseudocode describes how the repair operator works.

```
Procedure: Repair Operator
Begin
  For i = 1 To chromosome_length - 1
    op_i ← operation at location i in the chromosome;
    pd_i ← predecessor of op_i in the task sequence;
    ex ← True
    Repeat While (pd_i ≠ ∅) And (ex ≔ True)
      For j = i + 1 To chromosome_length
        op_j ← operation at location j in the chromosome;
        If (pd_i ≔ op_j) Then
          ex ← True;
          Exchange locations of op_i and op_j;
          Update op_i at location i and pd_i of op_i;
          Exit For
        Else ex ← False;
      End
  End
End
```

3.6 Termination Criteria

Termination criteria are used to determine when the GA-based heuristic should be stopped. Note that making decision on which sequence mobile robots and machines should handle tasks is a part of real-time activities of production planners. Therefore, on the one hand, if the best solutions over generations do not converge to a value, the maximum generation G_m would be used to stop the run. On the other hand, if the best solution does not improve over G_c consecutive generations, it would not be valuable to continue searching.

4 Numerical Example

In this section, an example problem is generated to examine performance of the GA-based heuristic. An FMS in the example problem has three machines M1, M2, and M3. There independent tasks with a total of seven operations are to be carried out. Two identical mobile robots are used to process the operations on three machines. The processing times of the operations are given in Table 1. This table also gives the precedence constraints among the operations in each task, e.g., the second operation of task 1 can be carried out only after the first operation of task 1 is complete. The traveling times of mobile robots from one machine to another are given in Table 2.

Table 1. Processing time of operations of tasks

Task	Operation	Machine	Processing time
1	1	M1	30
	2	M3	42
2	1	M2	24
	2	M1	18
	3	M3	36
3	1	M2	30
	2	M3	24

Table 2. Traveling time of robots from one machine to another

From/To	M1	M2	M3
M1	0	12	12
M2	16	0	20
M3	12	12	0

For GA parameters, the population size of 50 is used and probabilities of crossover P_c and mutation P_m are set to be 0.6 and 0.1, respectively. The termination is stop at the generation G_m of 200 or if no improvement is made after G_c of 50 generations. The proposed heuristic has been programmed in VB.NET and run on a PC having an Intel® Core i5 2.67 GHz processor and 4GB RAM. Fig. 4 shows the convergence of the best solution of the proposed heuristic.

The best solution obtained is given as: 21,2 - 11,1 - 12,1 - 31,2 - 22,2 - 23,2 - 32,1. The time required to complete all tasks or makespan is 160 time units and the computation time in this case is less than a second. The best solution of the example problem is depicted graphically by a Gantt chart with a small modification made to represent the schedules of the mobile robots. As shown in Fig. 5, a row is added to each mobile robot in order to show time intervals during which a mobile robot is transporting and performing its assigned operations. These time intervals are illustrated by non-colored bars and colored bars for transporting and performing the assigned operations to/at the destination machines, respectively. Note that the time interval needed to complete transporting may include part for an empty trip depending on the previous location of the mobile robot. This part is represented in the Gantt chart by using a shaded area in the non-colored bar, e.g., operation 32 is transported from machine 2 to machine 3 by mobile robot 1 at machine 3.

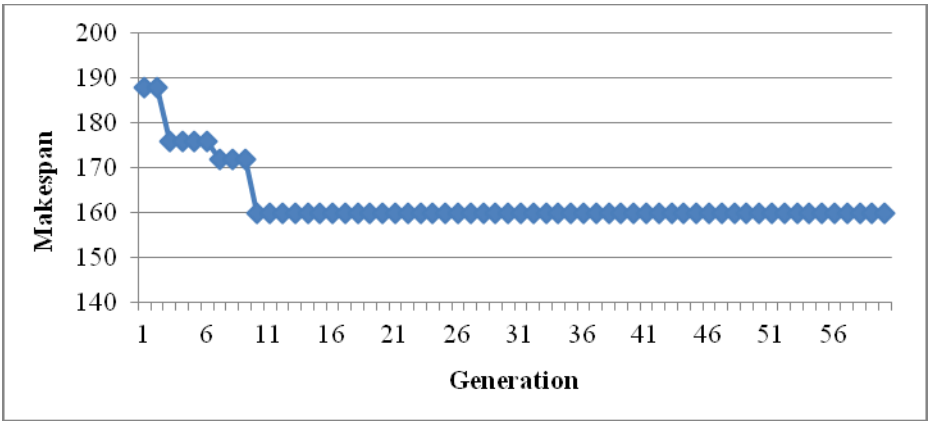

Fig. 4. Convergence of the GA-based heuristic

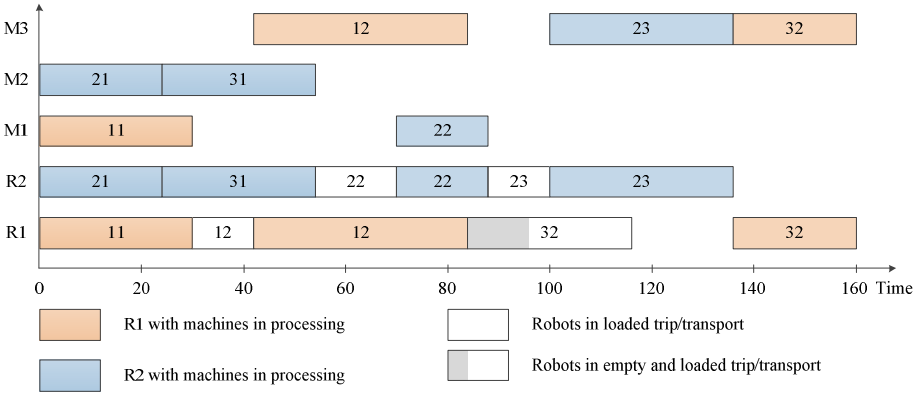

Fig. 5. Gantt chart for the best solution of the example problem

5 Implementation of Proposed Approach in Multi-agent System

This section generally describes how the proposed approach has been implemented in a multi-agent system and highlights the interactions between difference components. The proposed genetic algorithm-based heuristic, the decision-making nucleus, is integrated into the Mission Planning (MP) module of the Mission Planner and Controller (MPC). At first, the input data is derived from an operator through a user interface provided by the MPC or from Manufacturing Execution System (MES). Then, the MP module uses the input data to generate the best schedule/plan which serves as an input to the Mission Control (MC) module of the MPC (this best schedule is also shown to the operator via the user interface). Next, the MC module uses XML-based TCP/IP communication via a wireless network to command mobile robots to execute tasks in succession according to the best schedule and feedback from these mobile robots. In practice, there might be some unexpected events such as breakdown of machines or

mobile robots. These events will be reported by MES or the mobile robots so that the MC module is able to update current status of the shop floor and then call the MP module to reschedule the machines and mobile robots if needed. The MP module will in turn use the current status as new input, re-optimize to find an alternative schedule, and send this schedule back to the MC module for execution. Fig. 6 below depicts the aforementioned multi-agent system architecture.

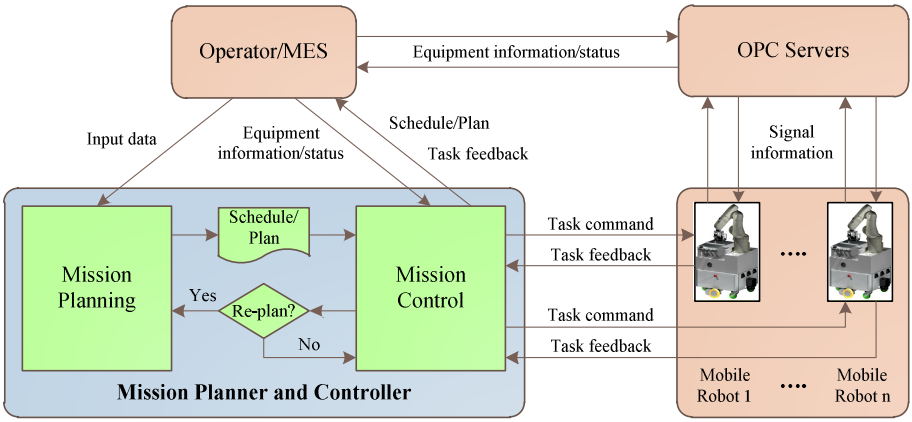

Fig. 6. Multi-agent system architecture

6 Conclusions

In this paper, a problem of simultaneous scheduling of machines and autonomous mobile robots in an FMS environment is studied. To complete all tasks in minimum possible time, it is important for production planners to determine in which sequence the mobile robots and machines should transport and process operations of tasks while satisfying a number of practical constraints. The main novelty of this research lies in the consideration of the participation in performing tasks at machines of the mobile robots. A genetic algorithm-based heuristic was developed to find the best solutions for the problem. An example problem was generated to demonstrate the efficiency of the proposed heuristic. The result showed that the proposed heuristic was significantly fast to obtain the best solution. The solution is useful for decision making at operational levels and the proposed approach provides a solid framework that enables to model and evaluate scheduling tasks of multi-agent systems. The implementation of the proposed approach and its interaction with other components in a multi-agent system was also generally described and highlighted. For further research, lot-sizing procedures for parts with respect to the capacity of the mobile robots should be considered as integral part of the optimization process. Furthermore, rescheduling mechanisms based on obtained schedules and feedback from the mobile robot fleet and machines should be also developed to deal with real-time disturbances.

Acknowledgments. This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

References

1. Abdelmaguid, T.F., Nassef, A.O., Kamal, B.A., Hassan, M.F.: A Hybrid GA/Heuristic Approach to the Simultaneous Scheduling of Machines and Automated Guided Vehicles. Int. J. Prod. Res. 42, 267–281 (2004)
2. Bilge, Ü., Ulusoy, G.: A Time Window Approach to Simultaneous Scheduling of Machines and Material Handling System in an FMS. Oper. Res. 43, 1058–1070 (1995)
3. Bocewicz, G., Wójcik, R., Banaszak, Z.: AGVs Distributed Control Subject to Imprecise Operation Times. In: Nguyen, N.T., Jo, G.-S., Howlett, R.J., Jain, L.C. (eds.) KES-AMSTA 2008. LNCS (LNAI), vol. 4953, pp. 421–430. Springer, Heidelberg (2008)
4. Bocewicz, G., Wójcik, R., Banaszak, Z.: On Undecidability of Cyclic Scheduling Problems. In: Karagiannis, D., Jin, Z. (eds.) KSEM 2009. LNCS (LNAI), vol. 5914, pp. 310–321. Springer, Heidelberg (2009)
5. Bocewicz, G., Banaszak, Z.: Declarative Modeling of Multimodal Cyclic Processes. In: Golinska, P., Fertsch, M., Marx-Gómez, J. (eds.) Information Technologies in Environmental Engineering. Environmental Science and Engineering – Environmental Engineering, vol. 3, pp. 551–568. Springer, Heidelberg (2011)
6. Dang, Q.-V., Nielsen, I.E., Bocewicz, G.: A Genetic Algorithm-Based Heuristic for Part-Feeding Mobile Robot Scheduling Problem. In: Rodríguez, J.M.C., Pérez, J.B., Golinska, P., Giroux, S., Corchuelo, R. (eds.) Trends in PAAMS. AISC, vol. 157, pp. 85–92. Springer, Heidelberg (2012)
7. Dang, Q.V., Nielsen, I., Steger-Jensen, K., Madsen, O.: Scheduling a Single Mobile Robot for Part-feeding Tasks of Production Lines. J. Intell. Manuf. (2012), doi:10.1007/s10845-013-0729-y
8. Deroussi, L., Gourgand, M., Tchernev, N.: A Simple Metaheuristic Approach to the Simultaneous Scheduling of Machines and Automated Guided Vehicles. Int. J. Prod. Res. 46, 2143–2164 (2008)
9. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York (1989)
10. Jerald, J., Asokan, P., Saravanan, R., Delphin Carolina Rani, A.: Simultaneous Scheduling of Parts and Automated Guided Vehicles in an FMS Environment Using Adaptive Genetic Algorithm. Int. J. Adv. Manuf. Technol. 29, 584–589 (2006)
11. Lin, L., Shinn, S.W., Gen, M., Hwang, H.: Network Model and Effective Evolutionary Approach for AGV Dispatching in Manufacturing System. J. Intell. Manuf. 17, 465–477 (2006)
12. Reddy, B.S.P., Rao, C.S.P.: A Hybrid Multi-Objective GA for Simultaneous Scheduling of Machines and AGVs in FMS. Int. J. Adv. Manuf. Technol. 31, 602–613 (2006)
13. Ulusoy, G., Bilge, Ü.: Simultaneous Scheduling of Machines and Automated Guided Vehicles. Int. J. Prod. Res. 31, 2857–2873 (1993)
14. Ulusoy, G., Sivrikaya-Şerifoğlu, F., Bilge, Ü.: A Genetic Algorithm Approach to the Simultaneous Scheduling of Machines and Automated Guided Vehicles. Comput. Oper. Res. 24, 335–351 (1997)

Paper K

Scheduling of machines and mobile robots in FMS

This paper has been submitted to The International Journal of Advanced Manufacturing Technology

Scheduling of Machines and Mobile Robots in FMS

Quang-Vinh Dang, Izabela Nielsen

Department of Mechanical and Manufacturing Engineering, Aalborg University, Fibigerstræde 16, DK 9220 Aalborg East, Denmark

vinhise@m-tech.aau.dk, izabela@m-tech.aau.dk

Abstract. This paper deals with the problem of simultaneous scheduling of machines and autonomous mobile robots in a flexible manufacturing system (FMS). Two types of tasks, namely non-preemptive and preemptive tasks, are considered to be scheduled on flexible machines. The mobile robots in this study have the capability of not only transporting non-preemptive tasks between some machines similar to material handling devices but also processing preemptive tasks on other machines by using their manipulation arms. These mobile robots are allowed to interrupt their preemptive tasks to carry out transportation for non-preemptive tasks when needed. The performance criterion is to minimize the makespan. A genetic algorithm-based heuristic is presented which results in a significant increase in the speed of finding near-optimal solutions. To evaluate the performance of the genetic algorithm-based heuristic, a mixed-integer programming model has been developed for the problem. A numerical example and computational experiments are conducted to demonstrate the effectiveness of the proposed approach.

Keywords: scheduling; preemption; mobile robots; genetic algorithm; FMS

1. Introduction

The automation technology in combination with advances in production management has dramatically changed the equipment used by manufacturing companies as well as the issues in planning and control. These changes have led to an enormous increase in efficiency and flexibility so that the progress of automation has become a necessity in global competition. Consequently, highly automated and unmanned production systems have become more popular in several industrial sectors, e.g. automotive, chemical, robot, and pump manufacturing [5]. Typically, an automatic production system consists of intelligent and flexible machines grouped into cells in such a way that entire production of each part/product can be performed within one of the cells. Within a cell, material handling could be performed by automatic guided vehicles (AGVs), industrial (fixed) robots, and autonomous mobile robots. Mobile robot is a term

used to refer to robotic systems consisting of a robot arm mounted on a mobile platform, which allow performance of tasks that require both locomotion and manipulation abilities [12]. Similar to AGVs, mobile robots have the capability of moving around within their environment to transport a variety of part types from one point to another without human intervention. However, mobile robots are more significantly advanced than AGVs in terms of the capability to perform various value-added tasks on different machines or workstations based on their manipulation arms. The tasks include such processes as machine tending, pre-assembly, and quality inspection. In contrast to dedicated and/or fixed robots, mobile robots are able to bring both task flexibility and robotic mobility to industrial applications. Furthermore, using mobile robots can lead to production efficiency gains such as less energy usage or lower tool-changing costs than dedicated and/or fixed robots. These superior abilities in comparison with AGVs and fixed robots pave the way for autonomous mobile robots to find increasing applications in today manufacturing.

In this paper, a problem of simultaneous scheduling of machines and mobile robots in a flexible manufacturing system (FMS) is taken into account. The FMS consists of a number of operations of different tasks processed on a set of flexible machines, and a set of mobile robots. Some of the tasks are non-preemptive while the others are preemptive. The non-preemptive tasks require the mobile robots to transport material/parts between the machines. The preemptive tasks are particularly processed based on the interaction between the machines and mobile robots. In other words, the mobile robots operate the machines to execute these preemptive tasks. When a mobile robot is being occupied by a preemptive task and a non-preemptive task invokes transportation at some point in the scheduling period, this mobile robot will pause to process the preemptive task, do the transportation for the non-preemptive task before going back to processing the preemptive task if the preemptive task has not been finished yet. In that context, to utilize the system in an efficient manner requires the capability of properly scheduling and routing mobile robots in connection with scheduling different tasks on machines. This problem could be considered as a variant of the problem of simultaneous scheduling of machines and AGVs in which its subproblems, machine scheduling and AGV scheduling, are both known to be NP-hard [8]. The main novelties of this research are that mobile robots are responsible for not only

transporting non-preemptive tasks but also processing preemptive tasks, and mobile robots must interrupt their processing tasks to do transportation for non-preemptive tasks when needed. The objective is to minimize the makespan C_{max}, i.e. the time required to complete all tasks. Note that making decision in scheduling and routing mobile robots in connection with scheduling machines is part of real-time operations. This gives the added requirement that the best solutions must be obtained quickly. Furthermore, the complexity of the considered problem rapidly rises when the number of tasks and/or mobile robots increases. Therefore, in this paper focus is on developing a computationally efficient approach, namely GA-based heuristic for the simultaneous scheduling machines and mobile robots. To evaluate the performance of the proposed heuristics, a mixed-integer programming (MIP) model is also presented.

The remainder of this paper is organized as follows. In the next section, the literature survey of the research is described. The problem description with a mixed-integer programming model of the problem is presented in Section 3. The solution approach using a GA-based heuristic is developed in Section 4. Section 5 illustrates the results of the proposed heuristic and compares its performance with that of the MIP model. More computational experiments are further investigated in this section. Finally, conclusions and future research directions are drawn in Section 6.

2. Survey of literature

The problem of simultaneous scheduling of machines and mobile robots in an FMS has been modeled in several respects comparable to the joint scheduling problems of machines and AGVs. However, it is different from the problems concerning AGVs in the sense that mobile robots are capable of performing manufacturing/production tasks in the shop floor. Several approaches and models for exact or (meta-) heuristic algorithms have been proposed to address problems of this type. Exact methods are mainly used for the research of simple or particular FMS. Blazewicz et al. [3] study the model of an FMS taking into account both machine and vehicle scheduling, and then propose a dynamic programming approach to construct optimal production and vehicle schedules. This FMS is later formulated in mixed-integer programming by Bilge and Ulusoy [2]. According to the authors, the resulting model is intractable in practice due to

its nonlinearity and its size. Khayat et al. [14] propose an integrated formulation of the combined production and material handling scheduling problems using mathematical programming and constraint programming techniques. Caumond et al. [4] deal with the linear formulation of an FMS considering the maximum number of jobs, limited input/output buffer capacities, empty-vehicle trips and no-move-ahead trips concurrently. Nevertheless, only one AGV is considered as a special case of the general FMS.

Heuristic methods are well adapted to study most of the FMS. On the one hand, some works are dedicated to simplified forms of the material handling system of the FMS considering only one transport device. As illustration, Soylu et al. [19], Hurink and Knust [11], Lacomme et al. [15] propose, respectively, neural network, tabu search, and heuristic branch-and-bound approaches for scheduling of the FMS based on a single AGV or transport robot. On the other hand, many works are undertaken on the FMS scheduling with multiple AGVs. Ulusoy and Bilge [20] and Bilge and Ulusoy [2] propose an iterative method based on the decomposition of the master problem into two sub-problems: machine scheduling and vehicle scheduling. A heuristic algorithm generates machine schedules to solve the first problem. A solution heuristic based on sliding-time-window approach is introduced to find feasible solutions to the vehicle scheduling problem given the machine schedules. Ulusoy et al. [21] deal with the problem of concurrent scheduling of machines and AGVs by proposing a genetic algorithm (GA) which provides a suitable coding scheme to represent both dimensions of the search space: operation sequencing and AGV assignment. Abdelmaguid et al. [1] introduce a hybrid method composed of a GA for scheduling of machines and a heuristic for scheduling of vehicles. Reddy and Rao [18] present a hybrid multi-objective GA to solve the simultaneous scheduling of machines and AGVs in an FMS in which makespan, flow time, and tardiness are performance criteria. Jerald et al. [13] address the problem of simultaneous scheduling of parts and AGVs in an FMS environment using adaptive GA. Lin et al. [17] model an AGV system by using network structure and propose an effective evolutionary approach for solving a kind of AGV problems. Deroussi et al. [8] describe an efficient neighboring system implemented into three different meta-heuristics and a new solution representation based on vehicles rather than machines. Lacomme et al. [16] introduce a framework based on a disjunctive graph to modelize the joint

scheduling problem and on a memetic algorithm for machines and identical AGVs scheduling.

Although much related research has been completed, the problem of simultaneous scheduling of machines and autonomous mobile robots in an FMS has not been studied in the literature. The considered mobile robots in this problem have the capability of not only transporting non-preemptive tasks between some machines similar to material handling devices but also processing preemptive tasks on other machines by using their manipulation arms. During operations, mobile robots are allowed to interrupt their processing tasks to do transportation for non-preemptive tasks when needed. In that context, the surveyed approach are not well suited and cannot be directly used to solve this problem due to the lack of a suitable mechanism for scheduling and routing mobile robots in relation to scheduling machines while taking into account precedence and routing constraints. Therefore, in this paper an MIP model is first formulated to find optimal solutions for the problem. However, the problem is composed of two interrelated decision problems that are machine scheduling and mobile robot scheduling. Both problems are known to be NP-hard, resulting in a more complicated NP-hard problem as they are considered simultaneously. Due to the intractability of NP-hard nature [9], the MIP model could only be applicable to small-scale problems in practice because its computation time significant increases as the problem size grows. Hence, in order to deal with large-scale applications, a heuristic based on GA, a promising algorithm to this class of problems, is then developed to find near-optimal solutions for the problem of simultaneous scheduling of machines and mobile robots. Compared to other optimization methods, the major benefit of GA regards multiple directional searches using a set/population of candidate solutions which enables GA to search in several directions concurrently. In this way, many paths to the optimum are processed in parallel, which leads to a clear improvement in performance. In addition, since information from many different regions is used, GA is resistant to remain trapped in a suboptimal solution and able to move away from it if the population finds better solutions in other areas [6]. Finally, the quality of the near-optimal solutions achieved by the proposed heuristic are compared and evaluated by using the MIP solutions as reference points in a numerical example and computation experiments.

3. Problem description

Flexible Manufacturing Systems are highly automated production systems capable of producing a variety of part/component types. An FMS originally includes intelligent and flexible machines, automated storage and retrieval systems, and material handling devices such as AGVs or robots. However, the development of automation technology has significantly changed the manufacturing equipment in the shop floor. With these changes, autonomous mobile robots have been designed and manufactured so that they can combine the flexibility of service robots, e.g. AGVs, with the efficiency of industrial robots, i.e. dedicated and fixed robots in industry. This enables these mobile robots not only to transport parts/components between machines but also to operate machines to process tasks. Therefore, they have been widely employed in not only small companies, which focus on exact applications and a smaller range of products, but also large companies, which can diversify their applications in a longer term and larger range. For instance in a pump parts manufacturing factory, a mobile robot is assigned to perform a pre-assembly task, and during that operation it also has to transport and feed material into some production lines when needed. In this study, the coordination between such mobile robots and machines in an FMS is considered. Fig. 1 below shows a typical layout of the FMS.

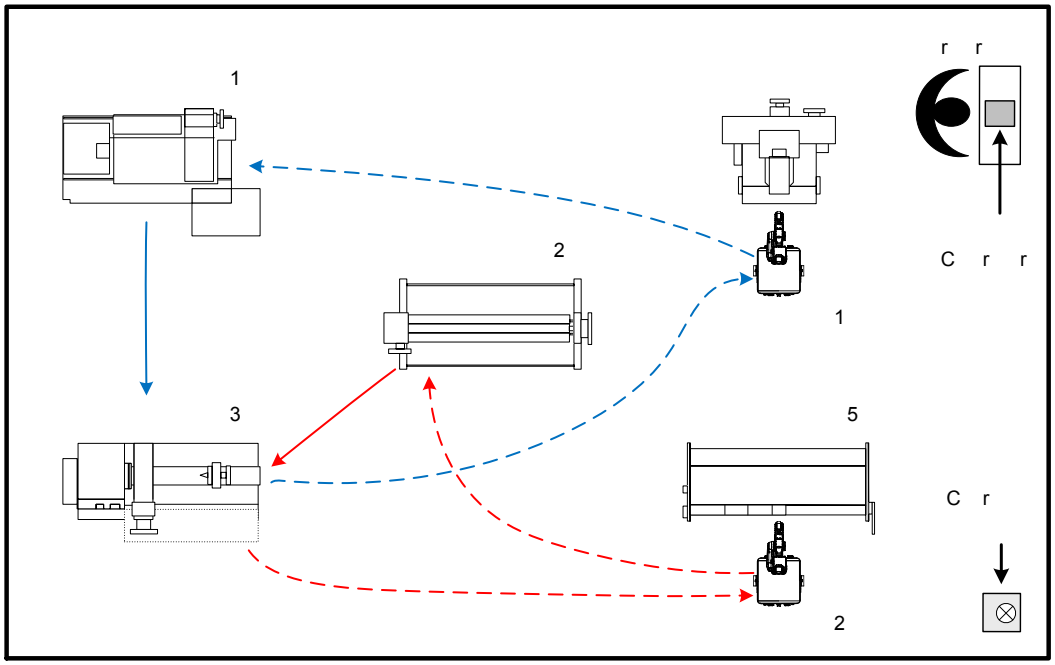

Fig. 1 Typical layout of the FMS with mobile robots

The FMS consists of a number of tasks processed on a set of machines, and a set of mobile robots. These tasks are classified into two types which are non-preemptive and preemptive. Each non-preemptive task consists of a set of operations that cannot be interrupted, i.e. each operation in this type must execute without interruption from its starting time to its ending time [22]. On the other hand, each preemptive task considered in this paper has only one operation that can be interrupted at any time to let some other operations execute. There is no restriction on the number of interruptions or on the duration of an interruption. During operations, the non-preemptive tasks require mobile robots to transport materials/parts between some machines while the preemptive tasks need the participation of the mobile robots in the processing on the other machines. Each mobile robot may carry out the transportation for different non-preemptive tasks, but it is assigned to process only one preemptive task on a specific machine. When being occupied by a preemptive task, a mobile robot may be invoked for transportation of a non-preemptive task at some points in the scheduling period. This mobile robot will pause to process the preemptive task, carry out the transportation for the non-preemptive task, and go back to processing the preemptive task if the preemptive task has not been finished yet. In practice, e.g. in a pump parts manufacturing factory, production operators may set the maximum number of operations of non-preemptive tasks which mobile robots can transport each time being away from their preemptive tasks. This prevents the mobile robots from leaving their preemptive tasks for a long period of time, which may lead to the cancellation of these tasks due to some practical issues in the shop floor. To some extent, this also helps to increase the utilization of these robots. Within the scope of this study, any mobile robot is set to come back to its processing machine after each achieved transportation. An example illustrating such preemption case is given in Fig. 2. The objective function is to find the best schedules which minimize the makespan, i.e. time required to complete all tasks.

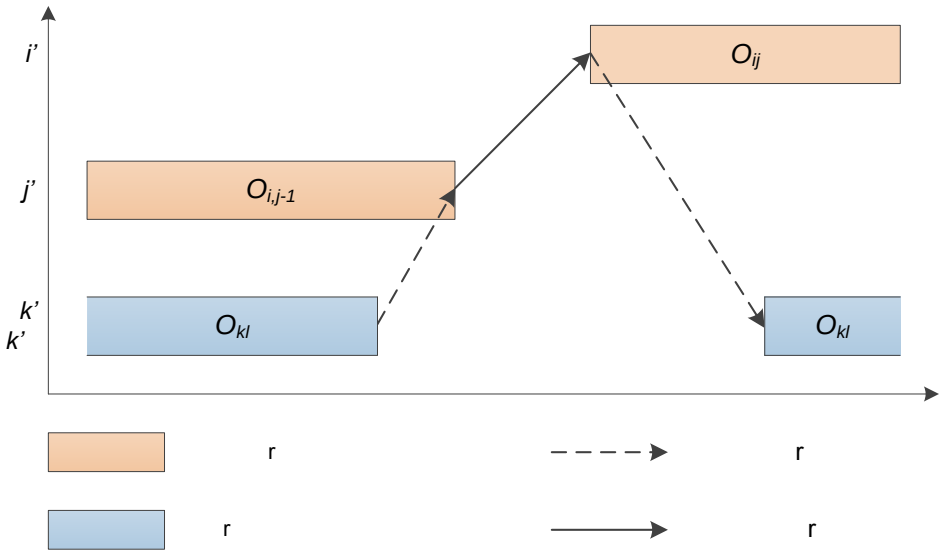

Fig. 2 Illustration of a preemption case

To enable the construction of a joint schedule of machines and mobile robots, the following assumptions are made:

- Each task is available at the beginning of the scheduling period.
- The first operation of each task is available at a machine at the beginning of the scheduling period.
- Each operation sequence of each task (the route of each part type) is available before making scheduling decisions.
- Each mobile robot can transport only one kind of parts at a time.
- There is sufficient input and output buffer space at each machine.
- Traveling time is only machine-dependent and deterministic. Loading and unloading time is included in the traveling time of loaded trips. Processing time is also deterministic.
- Such issues as traffic congestions, mobile robot collisions, machine failures, and scraps are not considered in this paper.

The joint machines and mobile robot scheduling problem is formulated as an MIP model. In addition to some basic operation and machine scheduling constraints, the MIP model introduces precedence constraints for transportation of non-preemptive tasks in relation to preemptive tasks, constraints for assignment of transportation to mobile robots, and constraints for preemptive tasks. The MIP model and notations are written in the following:

Notations

i, i', i^* : index of tasks

j, j', j^* : index of operations

k : index of mobile robots

N : set of non-preemptive tasks

P : set of preemptive tasks

M : set of machines

R : set of mobile robots

O_m : set of operations of non-preemptive tasks to be performed on machine m

o_{ij} : the j-th operation of task i

T_{ij} : transportation for operation o_{ij}

n_i : number of operations for task i

p_{ij} : processing time of operation o_{ij}

$t_{ij,i'j'}$: traveling time from machine of operation o_{ij} to machine of operation $o_{i'j'}$

Decision variables

T : time required to complete all tasks

s_{ij}^p : starting time of operation o_{ij}

s_{ij}^t : starting time of transportation T_{ij}

$$y_{iji'j'} = \begin{cases} 1 & \text{if operation } o_{ij} \text{ precedes operation } o_{i'j'} \\ 0 & \text{otherwise} \end{cases}$$

$$z_{iji'j'} = \begin{cases} 1 & \text{if transportation } T_{ij} \text{ precedes transportation } T_{i'j'} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if robot } k \text{ carries out transportation } T_{ij} \text{ for operation } o_{ij} \\ 0 & \text{otherwise} \end{cases}$$

Mixed-integer programming model

Objective function: minimize T (1)

Subject to:

$$s_{ij}^p - s_{i'j'}^p + Ly_{iji'j'} \geq p_{i'j'} \qquad \forall (i, j), (i', j') \in O_m, m \in M \tag{2}$$

$$s_{i'j'}^p - s_{ij}^p + L\left(1 - y_{iji'j'}\right) \geq p_{ij} \qquad \forall (i, j), (i', j') \in O_m, m \in M \tag{3}$$

$$s_{i'j'}^t - s_{ij}^t + L\left(1 - z_{iji'j'}\right) \geq t_{ij-1,ij} + t_{ij,i^*j^*} + t_{i^*j^*,i'j'} \tag{4}$$

$$\forall i, i' \in N,\ j = 2,...,n_i,\ j' = 2,...,n_{i'}, \forall i^* \in P,\ j^* = 1$$

$$z_{iji'j'} + z_{i'j'ij} = x_{ijk} x_{i'j'k} \qquad \forall i, i' \in N,\ j = 2,...,n_i,\ j' = 2,...,n_{i'} \tag{5}$$

$$s_{ij}^p + p_{ij} \le s_{i,j+1}^t \qquad \forall i \in N,\ j = 1,...,n_i - 1 \tag{6}$$

$$s_{ij}^t + t_{ij-1,ij} \le s_{ij}^p \qquad \forall i \in N,\ j = 2,...,n_i \tag{7}$$

$$\sum_{k \in R} x_{ijk} = 1 \qquad \forall i \in N,\ j = 1,...,n_i \tag{8}$$

$$s_{ij}^p + p_{ij} \le T \qquad \forall i \in N,\ j = 1,...,n_i \tag{9}$$

$$s_{i^*j^*}^p + p_{i^*j^*} + \sum_{i \in N} \sum_{j=2}^{n_i} x_{ijk} \left(t_{i^*j^*,ij-1} + t_{ij-1,ij} + t_{ij,i^*j^*} \right) \le T \qquad \forall i^* \in P,\ j^* = 1, k \in R \tag{10}$$

$$x_{ijk}, y_{iji'j'}, z_{iji'j'} \in \{0,1\} \qquad \forall i, i' \in N,\ j = 1,...,n_i,\ j' = 1,...,n_{i'}, k \in R \tag{11}$$

$$s_{ij}^p, s_{ij}^t \ge 0 \qquad \forall i \in N,\ j = 1,...,n_i \tag{12}$$

The objective function (1) minimizes the makespan. Constraints (2) and (3) represent the operation non-overlapping constraints which imply that a machine cannot process more than one operation at a time, where L is a very large number. Constraints (4) and (5) represent the mobile robot non-overlapping constraints which imply that a mobile robot cannot carry out more than one transportation task at a time. Constraint (6) ensures that the transportation for an operation could only start after the predecessor of that operation completes. Constraint (7) ensures that an operation could only start after the transportation for that operation finishes. Constraint (8) ensures that the transportation for an operation is carried by only one mobile robot. Constraints (9) and (10) ensure that completion time of non-preemptive operations (operations of non-preemptive tasks) and preemptive operations (operations of preemptive tasks) respectively cannot exceed the makespan T. Constraints (11) and (12) imply the types of variables.

The MIP model contains a number of decision variables that are constrained to have only integer values. Integer variables make optimization problems non-convex and thus far more difficult to solve. Memory and solution time may rise exponentially as the size of problem increases with more added integer variables. Therefore, in practice the MIP model could be applicable only to small-scale problems with a few tasks and/or few machines and/or few mobile robots. For these scenarios, the MIP model will give optimal solutions which

could be used as reference points to quantify the scale of benefits achieved by the heuristic based on GA which is developed in the next section.

4. Genetic algorithm-based heuristic

Ever since GA was introduced, it has emerged as one of the most broadly applicable and efficient search procedures for solving various combinatorial optimization problems. In general, a GA is referred to as a stochastic artificial intelligent technique whose solution search process mimics natural evolutionary phenomena [10]. In this section, GA is used to develop a heuristic which is allowed to convert the described problem to the way that near-optimal solutions could be found. The GA-based heuristic shown in Fig. 3 consists of the following main steps: genetic representation; initialization; decoding operator and fitness evaluation; genetic operators including crossover, mutation, and selection; reparation operator; termination criteria.

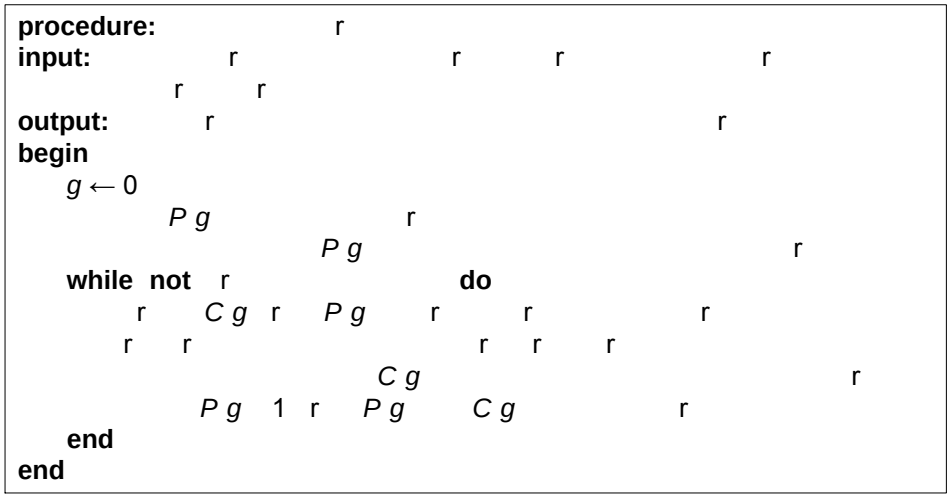

Fig. 3 Procedure of GA-based heuristic

4.1. Genetic representation

One of the main concerns when applying GAs to an optimization problem is to find an appropriate coding scheme that transforms feasible solutions into representations amenable to genetic search and reversibly decode these representations [17]. For the problem under consideration, a feasible solution can be encoded by a chromosome representing non-preemptive operation sequencing and mobile robot assignment. It means that each gene in the chromosome is made up of two parts. The first part refers to a non-preemptive operation on a specific

machine that is assumed to be scheduled at its earliest starting time. The second part identifies a mobile robot performing the transportation for that operation. In case the first part of a gene contains the first operation of a task, the second part of that gene will be zero (0) indicating that the first operation of the task does not need to be transported by a mobile robot. It is because the first operation of each task is assumed to be available at a machine at the beginning of the scheduling period as mentioned above. The chromosome length equals the total number of non-preemptive operations of all tasks. Fig. 4 below illustrates a feasible chromosome of an example problem with 5 operations of 2 tasks, 3 machines, and 2 mobile robots.

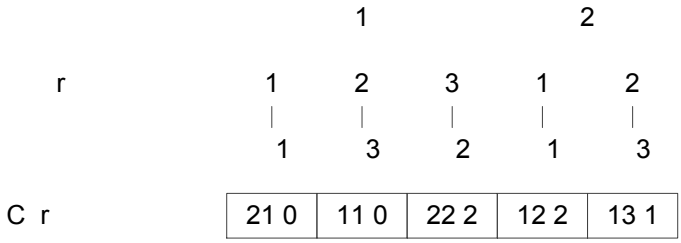

Fig. 4 Illustration of a feasible chromosome

4.2. Initialization

Random chromosomes are generated for providing solutions to the initial population. A chromosome is constructed of gene by gene. The first part of each gene is assigned an eligible operation (an operation is said to be eligible if all its predecessors are assigned). If that eligible operation is the first operation of any task, zero is assigned to the second part of the gene. Otherwise, one of the mobile robots is randomly chosen to complete the gene. If the chromosome is not yet complete, the eligible set of operations is updated and the process continues. Fig. 5 below shows the pseudocode of the initialization method.

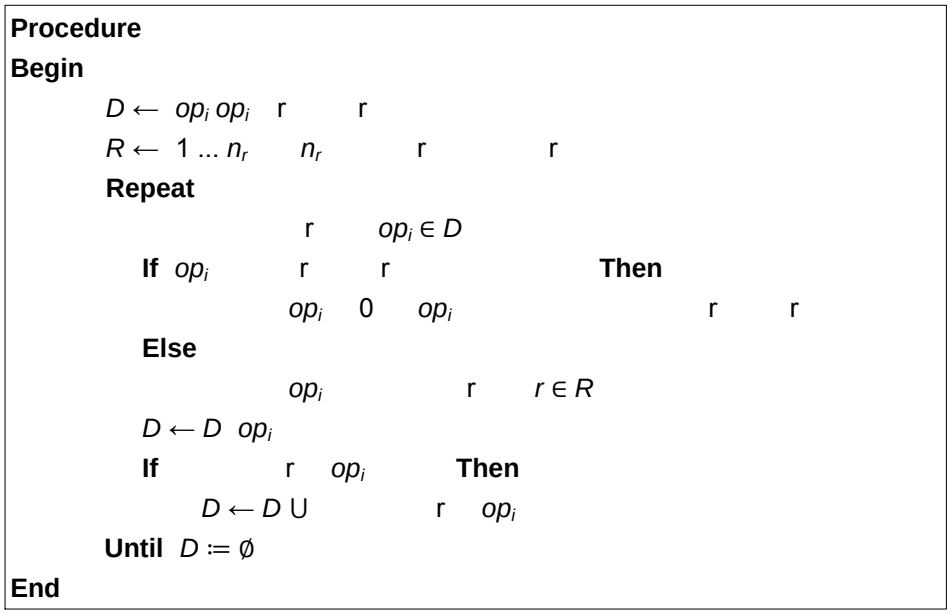

Fig. 5 Pseudocode of initialization method

4.3. Decoding scheme and fitness evaluation

After initialization or reparation routines, chromosomes are decoded and their fitness values are calculated. The decoding scheme is mainly composed of the decoding of operation scheduling and decoding of mobile robot assignment. Theoretically, the scheduling of operations is similar to job shop scheduling problem. Hence, the decoding of non-preemptive operation scheduling is carried out under consideration of the predecessor and the last operation processed on the predefined machine of each operation. Furthermore, before the processing of an operation can start, it has to be transported to the predefined machine by an assigned mobile robot. This invokes the decoding of mobile robot assignment. During this step, some information (e.g. total amount of processing time and/or completion time) of the preemptive operation performed by the mobile robot is also updated. In general, a schedule for preemptive operations is determined through the non-preemptive operation sequencing. Following the decoding scheme, the fitness evaluation will take place. The fitness value of a chromosome is equal to the maximum completion time of non-preemptive and preemptive operations. The decoding scheme and fitness evaluation are illustrated by the following pseudocode in Fig. 6 below.

```
Procedure                    F
Begin
    For i    1 To    r
        op ← operation at location i          r
        pd ← predecessor of op
        mc ← machine processing op
        mr ← mobile robot transporting op
        om ← operation processed by mr
        ns ←        r                 r          mc
        If  mr    0  Then    0            r              r r      r     op
            st ← 0
            If  ns    0  Then
                ls ←        r                  mc
                ct ← ls.
                If  st    ct  Then st ← ct
        Else
            st ← pd.
            If  ns    0  Then
                ls ←        r                  mc
                ct ← ls.
                If  st    ct  Then st ← ct
            ld ← mr.      r
            sr ←            r  mr  r          om
            md ←            r          pd
            d1, d2, d3 ←  r                    ld      sr, sr      md, md      mc
            If  mr.      r              d1    d2    pd.                Then
                mt ← pd.
                If  mr.        r            om.  r              Then
                    If  om.  r              – mr.        r
                    pd.                – mr.      r              d1    d2   Then
                        mr.        r          ← mr.        r
                        pd.                – mr.      r              d1    d2
                    Else
                        mr.        r          ← om.  r
                        om.                ← mr.      r              d1
                         om.  r              – mr.        r
            Else
                mt ← mr.      r              d1    d2;
            ft ← mt    d3
            If  st    ft  Then st ← ft
            mr.      r    destination ← mc
            mr.      r              ← ft
        op.    r          ← st
        op.                ← st    op.  r
                op              mc
    End
    For mr    1 To nr    nr        r          r
        If  mr.        r            om.  r              Then
            mr.        r      time ← om.  r
            om.            time ← mr.      r              d1    om.  r              –
            mr.        r
    End
    Cmax ←          op_i.              i    1...    r
                    om_j.              j    1..nr
End
```

Fig. 6 Pseudocode of decoding scheme and fitness evaluation

4.4. Genetic operators

Genetic operators mimic the process of heredity of genes to create new offspring at each generation. The operators, in essence, are used to alter the genetic composition of chromosomes and expected to yield improved offspring. Crossover, mutation, and selection are three main genetic operators.

Crossover operator generates offspring by combining the information contained in the parent chromosomes so that the offspring will have desirable features from their parents. The Roulette-wheel selection is used in the algorithm, which probabilistically selects the parent chromosomes based on their fitness values [7]. Owing to the nature of the considered minimization problem, the higher the makespan is, the less fitness a chromosome should show. Let $F(p)$ denote the fitness value under the solution represented by parent p, then $F'(p) = \max\{F(p)|1 \leq p \leq N_p\} - F(p)$ where N_p: population size. The expected probability of parent p to be selected is given by $F'_p/\sum F'_p$.

Several crossover operators have been proposed for permutation representation, e.g. two-point crossover, uniform crossover [1], partially-mapped crossover, order crossover, and position-based crossover [17]. Although the crossover operators may affect the efficiency of the search process, the quality of solutions is often reasonably close. In the presented experiment, a uniform crossover operating with probability P_c will be used to generate offspring as described below. Starting from the first operations on the parents, iteratively, one of the parents is randomly selected. The next unconsidered operation of the selected parent becomes the next operation on the first offspring while the next unconsidered operation of the other parent becomes the next operation of the second offspring. If the mobile robot selected for that operation is the same on both parents, then that selection is also made on the child; if not, one of the mobile robots of the parents is randomly chosen. Fig. 7 below depicts the uniform crossover.

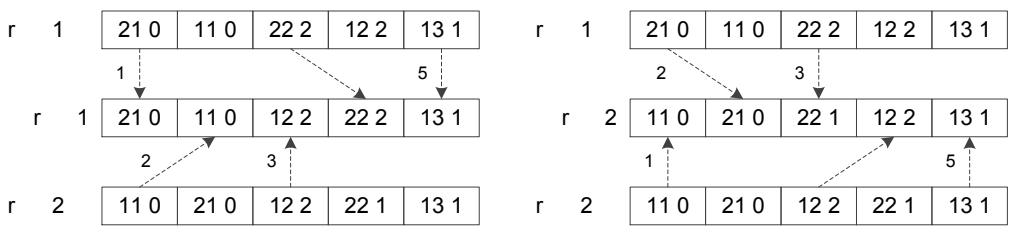

Fig. 7 Uniform crossover

Mutation operator produces spontaneous random changes in various chromosomes. For the current encoding method, there are two mutation operators, one for each part of a gene and with a probability P_m. The first mutation operator selects two random positions on a chromosome and swaps the operations with respect to those positions. Note that the chromosome may be infeasible in terms of precedence constraints after this mutation operation. Hence it has to be adjusted by using the reparation operator in Section 4.5. The second mutation operator replaces the mobile robot assignment at a gene with one of the mobile robots which is randomly chosen. This may lead to the same mobile robot assignment for a particular gene, and aim to prevent the loss of any good assignment.

Selection (reproduction) operator is intended to improve the average quality of the population by giving the high quality chromosomes a better chance to get copied into the next generation [17]. Various selection methods can be applied to this problem. In the presented experiment, (μ + λ) selection is used to choose chromosomes for reproduction. Under this method, μ parents and λ offspring compete for survival and the μ best out of the set of offspring and old parents, i.e. the μ lowest in term of the makespan, are selected as the parents of the next generation. This selection method guarantees that the best solutions up to now are always in the parent generation [7].

4.5. Reparation operator

A reparation operator is developed to validate chromosomes with any precedence violations after the mutation operator. This operator involves the exchange of locations of operations belonging to the same task such that a valid sequence of operations is achieved. Fig. 8 describes how the repair operator works.

```
Procedure        r
Begin
        For i    1 To    r                    – 1
            op_i ← operation at location i          r
            pd_i ←  r            r    op_i
            Repeat While  pd_i ≠ ∅
                  For j    i    1 To    r
                        op_j ←      r                  j          r
                        If  pd_i    op_j  Then
                                                    op_i       op_j
                           Exit For
                  End
        End
End
```

Fig. 8 Pseudocode of reparation operator

4.6. Termination criteria

Termination criteria are used to determine when the GA-based heuristic should be stopped. Note that making decision on which sequence mobile robots and machines should handle tasks is a part of real-time activities of production planners. Therefore, on the one hand, if the best solutions over generations do not converge, the maximum computation time CT_m would be used to stop the run. On the other hand, if the best solution does not improve over G_c consecutive generations, it would not be valuable to continue searching. The up-to-date best solution is then returned as the near-optimal solution. However, it should be noted that high-quality local optima might exist (in case of existing feasible solutions) because of the combinatorial nature of the problem [7].

5. Numerical experiments

To examine the performance of the MIP model and GA-based heuristic, a numerical example and computational experiments are conducted in this section. The numerical example has first been created to illustrate the results of both approaches. Various problem instances are then randomly generated and tested in order to provide more persuasive evidence of the performance of the GA-based heuristic. In the experiments, the MIP model has been coded and solved by the mathematical modeling language ILOG CPLEX, while the GA-based heuristic has been programmed in VB.NET. All the experiments run on a PC having an Intel® Core i5 2.67 GHz processor and 4 GB RAM.

5.1. Numerical example

An FMS considered in the numerical example consists of 5 tasks (3 non-preemptive and 2 preemptive tasks) with a total of 9 operations which are carried out on 5 flexible machines. Two autonomous mobile robots are employed in transporting the non-preemptive tasks between some machines and processing the preemptive tasks on the other machines. A layout for this example can be seen in Fig. 1. Table 1 gives the assigned machine numbers and processing time. This table also shows the precedence constraints among the operations in each task, e.g. the second operation of task 1 cannot be carried out before the first operation of task 1 is completed. The traveling time of the mobile robots from machines to machines are given in Table 2.

Table 1 Task description

Task	Operation	Machine	Mobile Robot	Processing Time (time unit)
1	11	M1	-	28
	12	M3	-	40
2	21	M2	-	32
	22	M1	-	26
	23	M3	-	42
3	31	M2	-	38
	32	M3	-	46
4	41	M4	R1	100
5	51	M5	R2	90

Table 2 Traveling time of robots from machines to machines (time unit)

From/To	M1	M2	M3	M4	M5
M1	0	8	8	10	10
M2	10	0	14	8	10
M3	8	12	0	10	12
M4	12	10	12	0	16
M5	10	10	14	10	0

To determine GA parameter settings, pilot runs are carried out to decide on the values of N_p, P_c, P_m, and G_c. Each parameter is tested at three different levels. These are respectively: N_p (50, 100, 200), P_c (0.4, 0.6, 0.8), P_m (0.05, 0.1, 0.2), G_c (50, 100, 200). There are ten observations under each level, and the 30 runs of each parameter are made in random. Furthermore, the time for each run is limited to *CTm* of 30 seconds (because making decision in simultaneous scheduling of machines and mobile robots is part of real-time operations in the shop floor as mentioned). The GA parameters are chosen according to the best results in terms of the objective value and computation time obtained with these pilot runs. They are as follows: 100, 0.8, 0.1, 100, and 30 for N_p, P_c, P_m, G_c, and CT_m, respectively.

For this numerical example, both MIP model and GA-based heuristic found the optimal solution for the problem. The time required to complete all tasks is 164, and the schedule is given as: 21,0 - 11,0 - 12,2 - 22,1 - 31,0 - 23,2 - 32,1. The proposed heuristic slightly faster obtains the optimal solution than the MIP model (0.1 second as opposed to 0.7 second). Fig. 9 shows the solution on Gantt chart. It can be seen from Fig. 9 that each mobile robot has to interrupt its preemptive task two times to carry out transportation for the non-preemptive tasks. For instance, mobile robot 2 interrupts task 5 the first time to transport task 1 from machine 1 to machine 3, and the second time to transport task 2 also from

machine 1 to machine 3. These interruptions consequently divide the duration of each preemptive task (or operation) into three separate parts as shown in the Gantt chart. In general, this numerical example has been illustrated the results of the proposed approaches. However, to make the evaluation more convincing, larger-sized problems will be investigated in the next section.

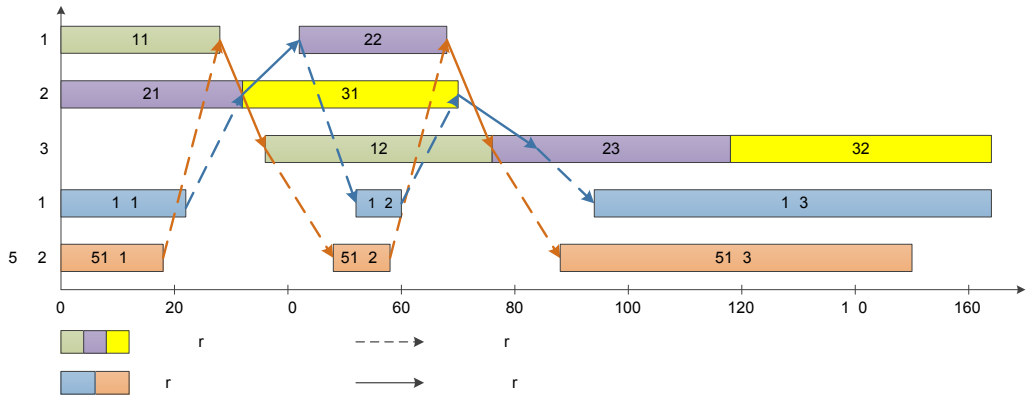

Fig. 9 Gantt chart for the optimal solution of the numerical example

5.2. Computational experiments

In this section, the performance of the proposed heuristic will be tested on a large number of problem instances. 20 problem instances are generated with different numbers of tasks, operations, machines, mobile robots and other system parameters. The number of all tasks and number of operations in each non-preemptive task are randomly generated in the ranges of [5, 20] and [2, 6], respectively. The number of machines and mobile robots are respectively distributed within the ranges of [5, 10] and [2, 4]. The processing time of non-preemptive and preemptive operations in time unit are respectively distributed within the ranges of [25, 50] and [100, 200] while the traveling time of mobile robots in time unit are generated in the interval [8, 18]. Note that the time/cost matrices of the generated traveling time should satisfy the triangle inequality. The problem sizes are shown in Table 3. The comparisons between the MIP and GA-based heuristic for 20 randomly generated problems are presented in Table 4. For each problem instance, the MIP is solved under consideration of the maximum computation time CT_m as the GA-based heuristic. The objective values and computation times of the proposed heuristic are the average of 30 runs.

Table 3 Problem sizes for 20 randomly generated problems

Problem instance	Problem size			Number of variables				Number of constraints
	Number of tasks	Number of operations	Number of machines	Number of robots	Integer variables	Continuous variables	Total variables	
1	5	12	5	2	74	22	96	154
2	7	25	6	2	412	48	460	839
3	6	16	5	2	144	30	174	296
4	8	30	7	2	592	58	650	1202
5	7	21	5	2	271	40	311	553
6	9	33	6	2	719	64	783	1457
7	10	38	8	2	957	74	1031	1936
8	11	44	7	3	1901	84	1985	3748
9	10	40	6	3	1626	76	1702	3204
10	12	48	7	3	2260	92	2352	4460
11	13	53	8	3	2736	102	2838	5405
12	14	62	7	3	4004	120	4124	7930
13	13	57	9	3	3235	110	3345	6399
14	15	68	8	3	4739	132	4871	9392
15	14	59	9	4	4462	112	4574	8743
16	17	81	10	4	8833	156	8989	17413
17	16	76	9	4	7880	146	8026	15524
18	19	94	10	4	12095	182	12277	23894
19	18	85	9	4	9805	164	9969	19343
20	20	100	10	4	13823	194	14017	27330

Table 4 Comparison between MIP and GA-based heuristic for 20 randomly generated problems

Problem instance	MIP		GA-based heuristic			
	Objective value	Computation time (s)	Objective value		Computation time (s)	
			Average	Standard deviation	Average	Standard deviation
1	248	1.76	248	0	0.25	0.03
2	691	30.00	474	8	0.40	0.08
3	306	5.38	309	2	0.29	0.04
4	818	30.00	534	6	0.46	0.06
5	537	30.00	399	3	0.33	0.05
6	932	30.00	591	10	0.59	0.13
7	1031	30.00	629	12	0.62	0.21
8	1379	30.00	678	18	0.93	0.18
9	1209	30.00	650	16	0.65	0.11
10	1544	30.00	702	18	1.04	0.22
11	1816	30.00	778	20	1.44	0.32
12	2043	30.00	963	24	1.92	0.33
13	1850	30.00	803	19	1.70	0.24

14	2207	30.00	975	24	2.18	0.47
15	1952	30.00	743	22	1.36	0.22
16	2876	30.00	1031	29	2.74	0.54
17	3032	30.00	1015	27	2.04	0.70
18	3836	30.00	1153	29	4.46	0.83
19	3050	30.00	1067	31	3.32	0.54
20	4282	30.00	1239	31	5.33	1.05

It can be observed from Table 4 that the GA-based heuristic is superior to the MIP for large problems. The MIP found feasible (not optimal) solutions within the time limit for 18 problem instances. Nevertheless, these solutions found by the MIP model are much worse than those found by the GA-based heuristic. Furthermore, the computation time shows that the proposed heuristic was significantly fast in obtaining the best solutions, e.g. approximately 5 seconds for problem instance 20 (the largest-sized problem). For the other problems, both MIP and GA-based heuristic found the optimal solution in problem instance 1 while the objective value found through the proposed heuristic are greater than that found by the MIP in problem instance 3. However, the difference is only about 1% and this is deemed to be an acceptable error. Furthermore, in terms of the objective value, the standard deviation is quite small in comparison with the average. These results provide more persuasive evidence to prove that the GA-based heuristic performs effectively.

6. Conclusions

This paper studies a novel problem of simultaneous scheduling of machines and autonomous mobile robots in an FMS. Two types of tasks which are preemptive and non-preemptive are taken into account. The mobile robots in this problem have capability of not only transporting non-preemptive tasks similar to material handling devices but also processing preemptive tasks by using their manipulation arms. The main novelty of this research lies in the fact that the mobile robot must interrupt their preemptive tasks to carry out transportation for non-preemptive tasks when needed. The objective function is to find the best schedules which minimize the time required to complete all tasks. This must be done while considering a number of technological constraints. A mixed-integer programming model to find exact optimal solutions for the problem was developed. Due to the NP-hard nature of the joint scheduling problem, in practice this solution approach

is only applicable to small-scale problems with a few tasks, machines, and mobile robots. A genetic algorithm-based heuristic was then proposed to find near-optimal solutions. The quality of these solutions could be then evaluated by using the MIP solutions as reference points to quantify the scale of benefits. A numerical example and computational experiments were presented to demonstrate the effectiveness of both approaches. The results showed that the proposed heuristic was significantly fast in obtaining near-optimal solutions. The results also provided persuasive evidence that the proposed heuristic is capable of solving problems of various sizes and more efficient than the MIP in terms of the objective value when giving the same maximum computation time. These solutions are useful to managers for decision making at operational levels and the proposed heuristic could be also applied in variety of tasks of not only mobile robots but also AGVs. For further research, a general model allowing mobile robots carrying out more than one transportation task before going back to their preemptive tasks should be taken into account. This also includes a rescheduling mechanisms based on the obtained schedules and feedback from the mobile robot fleet and shop floor. This will enable to deal with real-time disturbances such as machine or mobile robot breakdown.

Acknowledgements. This work has partly been supported by the European Commission under grant agreement number FP7-260026-TAPAS.

References

1. Abdelmaguid, T.F., Nassef, O.N., Kamal, B.A., Hassan, M.F., 2004. A hybrid GA/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles. Int. J. Prod. Res., 42 (2), 267–281.
2. Bilge, Ü., Ulusoy, G., 1995. A time window approach to simultaneous scheduling of machines and material handling system in a FMS. Oper. Res., 43 (6), 1058–1070.
3. Blazewicz, J., Eiselt, H.A., Finke, G., Laporte, G., Weglartz, J., 1991. Scheduling tasks and vehicles in a flexible manufacturing system. Int. J. Flex. Manuf. Sys., 4 (1), 5–16.
4. Caumond, A., Lacomme, P., Moukrim, A., Tchernev, N., 2009. A MILP for scheduling problems in an FMS with one vehicle. Eur. J. Oper. Res., 199 (3), 706–722.
5. Crama, Y., Kats, V., van de Klundert, J., Levner, E., 2000. Cyclic scheduling in robotic flow shops. Ann. Oper. Res., 96 (1–4), 97–124.
6. Dang, Q.V., Nielsen, I., 2013. Simultaneous scheduling of machines and mobile robots. In: Corchado, J.M., et al. (Eds.), PAAMS 2013 Workshops, CCIS 365, pp. 118–128.

7. Dang, Q.V., Nielsen, I., Steger-Jensen, K., Madsen, O., 2013. Scheduling a single mobile robot for part-feeding tasks of production lines. J. Intell. Manuf., doi: 10.1007/s10845-013-0729-y.
8. Deroussi, L., Gourgand, M., Tchernev, N., 2008. A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles. Int. J. Prod. Res., 46 (8), 2143–2164.
9. Ganesharajah, T., Hall, N.G., Sriskandarajah, C., 1998. Design and operational issues in AGV-served manufacturing systems. Ann. Oper. Res., 76, 109–154.
10. Goldberg, D.E., 1989. Genetic algorithms in search, optimization and machine learning. Kluwer Academic Publishers, Boston, MA.
11. Hurink, J.L., Knust, S., 2002. A tabu search algorithm for scheduling a single robot in a job-shop environment. Discrete Appl. Math., 119 (1–2), 181–203.
12. Hvilshøj, M., Bøgh, S., Nielsen, O.S., Madsen, M., 2012. Multiple part feeding – real-world application for mobile manipulators. Assembly Automation, 32 (1), 62–71.
13. Jerald, J., Asokan, P., Saravanan, R., Delphin Carolina Rani, A., 2006. Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithm. Int. J. Adv. Manuf. Technol., 29 (5–6), 584–589.
14. Khayat, G.E., Langevin, A., Riopel, D., 2006. Integrated production and material handling scheduling using mathematical programming and constraint programming. Eur. J. Oper. Res., 175 (3), 1818–1832.
15. Lacomme, P., Moukrim, A., Tchernev, N., 2005. Simultaneous job input sequencing and vehicle dispatching in a single-vehicle automated guided vehicle system: a heuristic branch-and-bound approach coupled with a discrete events simulation model. Int. J. Prod. Res., 43(9), 1911–1942.
16. Lacomme, P., Larabi, M., Tchernev, N., 2013. Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles. Int. J. Prod. Econ., 143 (1), 24–34.
17. Lin, L., Shinn, S.W., Gen, M., Hwang, H., 2006. Network model and effective evolutionary approach for AGV dispatching in manufacturing system. J. Intell. Manuf., 17 (4), 465–477.
18. Reddy, B.S.P., Rao, C.S.P., 2006. A hybrid multi-objective GA for simultaneous scheduling of machines and AGVs in FMS. Int. J. Adv. Manuf. Technol., 31 (5–6), 602–613.
19. Soylu, M., Özdemirel, N.E., Kayaligil, S., 2000. A self-organizing neural network approach for the single AGV routing problem. Eur. J. Oper. Res., 121 (1), 124–137.
20. Ulusoy, G., Bilge, Ü., 1993. Simultaneous scheduling of machines and automated guided vehicles. Int. J. Prod. Res., 31 (12), 2857–2873.
21. Ulusoy, G., Sivrikaya-Serifoglu, F., Bilge, Ü., 1997. A genetic algorithm approach to the simultaneous scheduling of stations and automated guided vehicles. Comput. Oper. Res., 24 (4), 335–351.
22. Yun, Y.S., 2002. Genetic algorithm with fuzzy logic controller for preemptive and non-preemptive job-shop scheduling problems. Comput. Ind. Eng., 43 (3), 623–644.