FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A Bias Random-key Genetic Algorithm for an AGV Scheduling Problem

Maria Sofia Freire Oliveira de Castro Ribeiro

FOR JURY EVALUATION

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: José Fernando Oliveira

January 25, 2016

Abstract

Nowadays, industrial environments commonly use autonomous agents. This is due to the need of boosting the productivity levels and increasing profits by reducing the production costs. Therefore, labour efforts and transportation time have to be brought to a minimum.

Solutions based on autonomous systems favor just-in-time delivery, a more accurate stock management and also a high level of efficiency, providing the possibility of working 24 hours a day without human intervention. AGVs are computer controled and so, they respond well in this type of systems. They are robust and flexible, contributing to the increase of the process performance. Moreover, AGVs can be used as means of transportation between worstations, aid in warehouse control and distribution.

However, to take full advantage of the AGVs features and to perform the several operations required, the shop-floor configuration needs more than one AGV. To do so, a reliable control system is of the outmost importance.

In this thesis, two algorithms were used. The routing algorithm is responsible for mapping the shop-floor and give valueable information to the scheduling algorithm that organizes the operations that need to be executed. The firs was provided by Follow Inspiration S.A. and the second is the main focus of this document.

BRKGA (Bias Random-key Genetic Algorithm) was the framework chosen to provide an answer for the scheduling problem. This algorithm uses random-keys to represent the chromosomes. A decoder is, initially, responsible for adjusting the results to a solution of the problem and, then, for calculating its' fitness value. The framework creates the first populations with the required chromosomes and at each iteration the solutions are sorted by their fitness value. The best ones go to the elite group and the others to the non-elite group. The next generation is composed by the selected elite elements, a new group of random-keys (mutants) and the result of the mating process between elite and non-elite solutions.

The results provided by the chosen approach show that the elitistic feature of the BRKGA is beneficial and this study is a good starting point for a solution to the industrial resource constrained project scheduling problem considered.

Resumo

Hoje em dia, agentes autónomos são comummente utilizados. Isto deve-se à necessidade de aumentar os níveis de produtividade e o lucro, reduzindo os custos de produção. Assim, o esforço laboral e o tempo de transporte têm de ser reduzidos em ambientes industriais.

Soluções baseadas em sistemas autónomos favorecem a entrega *just-in-time*, uma gestão de stocks mais precisa e também um alto nível de eficácia, uma vez que este tipo de soluções abre a possibilidade de ter um sistema que funcione 24 horas sem intervenção humana. Os AGVs são controlados por computador e, portanto, respondem bem a este tipo de sistemas. São robustos e flexíveis, o que contribui para o aumento do bom desempenho do processo. Além disso, os AGVs podem ser utilizados como meio de transporte entre postos de trabalho, como apoio no controlo de armazéns e na distribuição.

No entanto, para tirar proveito das vantagens da utilização de AGVs e para executar as diferentes operações, o chão de fábrica necessita mais do que um AGV. Para tal, um sistema de controlo fiável é da maior importância.

Nesta tese utilizam-se dois algoritmos. O algoritmo de *routing* é responsável por mapear o chão de fábrica e dar informações importantes ao algoritmo de escalonamento que organiza as operações a serem executadas. O primeiro foi cedido pela Follow Inspiration S.A. e o segundo é o principal foco deste documento.

BRKGA (Bias Random-Key Genetic Algorithm) foi a *framework* escolhida para obter melhores respostas para o problema de escalonamento. Este algoritmo usa chaves aleatórias para representar cromossomas. O *decoder* é, inicialmente, responsável por ajustar os resultados para uma solução do problema e, depois, para calcular o seu valor de *fitness*. A *framework* cria a população inicial com p cromossomas e a cada iteração as soluções são organizadas pelo seu valor de *fitness*. Os melhores vão para um grupo elite e os restantes para um grupo não-elite. A geração seguinte é composta pelos elementos elite selecionados, um novo grupo de chaves aleatórias (mutantes) e o resultado do processo de acoplamento entre soluções elite e não-elite.

Os resultados gerados por esta abordagem mostram que a característica elitista do BRKGA é benéfica e este estudo é um bom começo de uma solução para o problema industrial de escalonamento de projetos com recursos limitados considerado.

Agradecimentos

Primeiramente agradeço à minha família e, principalmente à minha mãe, por todo o apoio e carinho demonstrados desde o principio ao fim desta dissertação. Agradeço também a todos os meus amigos pelas brincadeiras, risos e momentos de descontração. Mas principalmente, à Mafalda Guimarães e ao Rafael Marques por serem as pessoas espetaculares que são e por terem sempre um conselho nas alturas mais difíceis, à Fátima Airosa por ser aquela pessoa que partilhou sempre comigo o esforço e que me manteve motivada para continuar e, por fim, mas não o menos especial, ao Bruno Martins por tanto me ter ajudado a todos os níveis, dentro e fora da faculdade, e nunca me ter deixado desistir.

Aos professores António Galrão Ramos e Luís Teixeira deixo também uma palavra de agradecimento por me terem ajudado em aspetos mais técnicos no que diz respeito a esta dissertação.

Ao meu orientador professor José Fernando Oliveira por ter prestado o apoio necessário para poder evoluir, sendo sempre prestável e direto em todas as suas observações. Além disso, o contacto com a Investigação Operacional, área lecionada por este professor e pela professora Maria Antónia Carravilha, despertou em mim a curiosidade de desenvolver competências nesta área da engenharia que era até então desconhecida.

Por último, agradeço à Follow Inspiration por me ter proporcionado esta oportunidade mas, principalmente, à Joana Santos por todos os esclarecimentos.

Maria Sofia Ribeiro

“And those who were seen dancing were thought to be insane
by those who could not hear the music”

Friedrich Nietzsche

Contents

List of Figures

List of Tables

Abreviaturas e Símbolos

AGV	Autonomous Guided Vehicle
BRKGA	Bias Random-Key Genetic Algorithm
GA	Genetic Algorithm
RCPSP	Resource Constrained Project Scheduling Problem
RKGA	Random-Key Genetic Algorithm
TEA*	Time-Enhanced A*

Chapter 1

Introduction

This document is a result of the development of a Master's degree in Electrotecnical and Computers Engineering thesis at Faculdade de Engenharia da Universidade do Porto.

This thesis was thought as an industrial solution for Follow Inspiration SA, and its' main goal was to come up with a way of using AGVs, a already developed routing algorithm and add a schedulling algorithm that would improve a production solution for future cooperations between Follow Inpiration SA and other companies, aiming for a better control of the Work-In-Progress in their facilities.

Production line schedulling problems are often studied and many techniques already exist to approach this problem, however not all are adaptable to any situation at hand and, so, a thorough analysis has been made to understand the limitations and requisitions that this project demands.

1.1 Automated Guided Vehicles

The first time Automated Guided Vehicles (AGVs) were used for industrial purposes was in 1955. "An Automated Guided Vehicle (AGV) is an automated guided cart that follows a guided path". Initially AGVs were used for physical distribution in manufacturing systems but their possibilities for other uses was noticed and taken advantage of. Nowadays they can follow motion, avoid obstacles and can perform tasks even without a determined path. Basically AGVs are a mean of transportation for materials without being driven manually by man. Although AGVs are a big asset for industrial facilities today, they are restricted by the type of materials that need transportation. AGVs are mostly used in four main areas:

1. Supply and disposal at production areas
2. Assembly plataforms
3. Retrieval (wholesale trade)
4. Supply and disposal (for example: Hospitals)

To ease the production process, facilities use more than one AGV thus having an Autonomous System. Autonomous Systems are hard to control because they demand a deep knowledge on how AGVs work and demand a routing system capable of maintaining the feasibility and security of the system.

1.2 Follow Inspiration

Follow Inspiration SA was founded in 2012 and develops software and hardware with a wide range of applications. Their main goal is to provide robotic navigation solutions with characteristics such as maping, image recognition and artificial intelligence to improve people's life.

To be able to provide complete solutions, Follow Inspiration as a multi-disciplinary team with knowledge in areas like management, marketing, computer science and engineering.

Follow Inspiration was awarded nationaly and internationaly for their leadind developments on technological solutions that can improve people's life.

1.3 Motivation and Methodology

As technology advances, companies feel the need to update and improve their assets and processes in order to be competitive and accompany market trends. Autonomous Systems are a big part of the tremendous changes that are being made in the industry today.

This thesis provides an opportunity to participate in something that can open a new perspective to the Follow Inspiration business model. The main goal of the development of this thesis was to come up with a task and operation oriented schedulling method, using multiple AGVs.

The schedulling algorithm organizes tasks by taking into account the total execution time of the whole process and the number of AGV stops.

1.4 Thesis Structure

After this introduction, Chapter 2 provides information about the tecnologies and concepts used in this thesis, and discusses their importance to the problem itself. Here, the core elements of this thesis are explained and some examples of their use are given.

Chapter 3 explains how the solution was built and how it works. Additionally, the variables used are defined and their role is explained. The results from this approach are analysed and compared in detail in Chapter 4.

The final chapter finalizes this work by giving a overall view of the thesis and also some recommendations for future work.

Chapter 2

State of the Art

This chapter is the result of the investigation made to develop a good foundation for the project at hand.

2.1 Autonomous Guided Vehicles

Due to the constant need of changing and adapting production, in order to keep up with the evolving market demands, fully automated systems were introduced as an alternative for manual or mechanical solutions.

AGVs are known for their flexibility, adaptability and robustness in industrial environments. Capable of transporting materials between worstations, they increase the efficiency and effectiveness of the production process [3]. Moreover, AGVs are easily adaptable without any or little modifications to the shop-floor's structure, easily traceable and work for unlimited time with few or no supervision [1].

Fully automated systems using AGVs are computer controlled, thus allowing a stable production rate, preventing quality issues, costs caused by reworks and providing a more secure production environment [1]. An example of an industrial AGV used is represented in Figure 2.1.

The efficiency of AGV use in manufacturing management was also the main focus of Robot@Factory for the 11th Portuguese Robotics Open. [4] describes the experience in the competition and mentions that the path planning and schedulling solution was their main focus. To aid in the preparation of the participants, the organization provided a simulator that represents the real environment. The simulated factory had two warehouses (input and output), several workstations and AGVs, responsible for the transportation between workstation and warehouses.

To have an efficient production plan is a huge challenge for companies and it is even more challenging to be able to supervise and control all the resources and operations at the same time. This is aknowledged in [4]. It is also why the use of an AGV based automated system capable of efficiently control an entire production process, was a substancial motivation for the work conducted in this article.

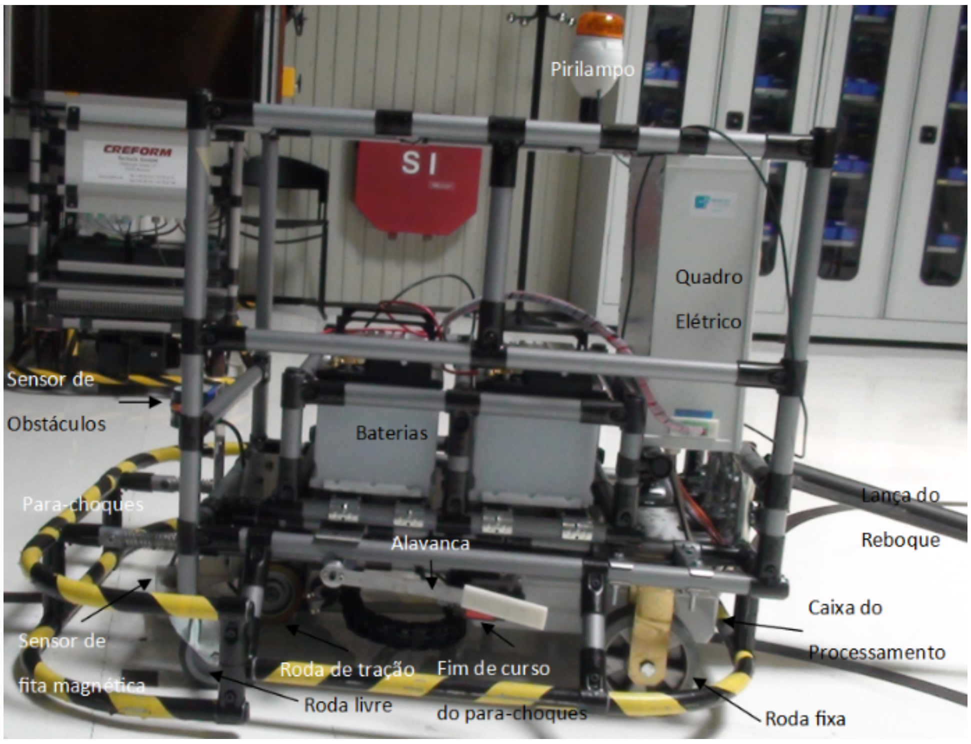


Figure 2.1: AGV from Volkswagen Autoeuropa [1]

The work portrayed by the authors included an optimization model that considered important aspects like the transportation and processing time, and then calculates a cost function used to determine which task to perform in a given instant of time.

The result of this article work was a succefully working transportation architecture for AGVs using a decentralized schedulling algorithm. This approach contributes to the idea that AGV based production systems are efficient in terms of task schedulling, improve processes, reduce production time and favour infrastructure management.

In [1] a low-cost AGV prototype system was tested as an alternative for the already existing hybrid system of Volkswagen Autoeuropa. Additionally, the company had a few sections of the factory using different types of AGVs, with different hardware and software, making the communication between AGVs more difficult and, in case of mafunction, the AGVs where hard to substitute by another without compromising the whole routing system. So, the solution proposed consists in a standardized AGV based system capable of transporting materials to the production line. This was then tested, taking into account some specific technical requirements, for different routes of the Autoeuropa factory.

The standardized system proposed by Eduardo Santos thesis turned out to be a possible alternative to the existing system capable of executing all tasks, meeting all the features imposed and being easily adapted according to the productions' needs. Moreover, some tests were conducted to evaluate this method, namely what concerns the maximum time the AGVs could work continuously and the maximum weight carried by them. The AGVs proved to be very robust, they were capable of working 16 hours and carrying 300kg, exceeding the lines' needs. It was also proven that the return of the investment on the new AGV based solution is fulfilled in a short period of time. Thus providing proof of the benefits of having a fully automated and standardized system using AGVs.

2.2 Time Enhanced A*

Time Enhanced A* (TEA*) is the search method used as a routing algorithm in an industrial environment using AGVs at [3]. This method is based on another called A*.

An A* search method uses a tiled or cell based grid map with two dimensions such as Figure 2.2 suggests.

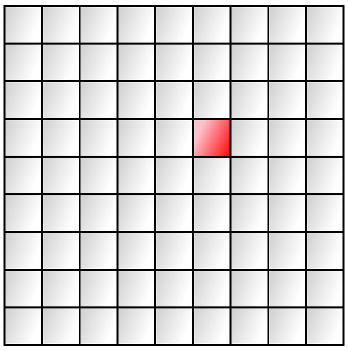


Figure 2.2: Cell based grid map [2]

In this search method, the movements are made through the tiles, edges or vertices of the grid and each movement made on this map has an associated cost. The most commonly used type of movement is between neighbor tiles, where the cost is calculated from and to the center point of each one of the tiles.

A* can calculate the path with the minimum cost from the starting point to the target by searching routes favouring the distance to the starting point, the distance to the end point, considering the already travelled distance. The A* grid map is also able to identify trajectory obstacles to prevent colisions ([2]). This is a flexible path-planning search method and it's used in various contexts such as game programming and robot real-time applications. An example of its use is described in [5].

TEA* uses a third dimension when comparing to the A* search method. In addition to the position in the map it also includes a representation of time. This third dimension can be seen as layers of the same grid-map considering the objects positions in the map and the changes that occur. So, TEA* recalculates the objects' routes by updating, in each iteration, their new positions on the map.

In [3] a multi-AGV system is considered and the temporal layers benefits are used to avoid and predict collisions and deadlocks, and efficiently process all tasks when confronted with a multi-agent system.

In industrial environments with several workstations and tasks, multi-AGV systems can be challenging to apply. Being capable of predicting the position of each AGV is a big and necessary part of that challenge. Therefore, the temporal layer added by the TEA* approach is a key component to calculate the position of each vehicle accurately, as well as preventing collision and deadlock situations.

In comparison to the A* search method, with only one AGV, during path planning, the cost function is always considered when analysing the next neighbor point. This function is calculated by adding the distance between the point that is being considered and the initial point, and by adding the distance to the final point.

Due to the temporal dimension, the authors are able to add differentiating characteristics to the traditional A* search method that aid in the development of a novel robot coordination system. Additionally, to efficiently ensure the feasibility of this system, the TEA* approach considers that the previous calculated path is a moving obstacle. Moreover, this feature is said to be a substancial contribution to the Robotic community that study and develop solutions for industrial application.

The authors also state that the TEA* method provides a significantly important step towards a functional multi-AGV autonomous industrial system.

2.3 Biased random-key genetic algorithms

Many real-life problems like routing, schedulling, inventory control, production and location can be represented by combinatorial optimization models. Heuristics are often used to aid in producing methods that provide good solutions to this type of problems quickly.

Heuristics are problem-dependent methods, so, they are built to take full advantage of the specificities of the problem at hand. Even though many problems can be solved by simple heuristics, more complex problems demand better solutions than the ones provided by simple them. Therefore, heuristics have been evolving throughout the last decades and benefiting from the guidelines provided by metaheuristics ([6] and [7]).

Metaheuristics are problem-independent and can use and combine simple heuristics that can explore the problems' search space and provide a better quality solution than those found by simpler optimization algorithms ([6]).

There are many metaheuristics commonly used for combinatorial optimization problems. Tabu search, iterated local search, swarm intelligence optimization, genetic algorithms, are some of the examples that can be found at [8].

In this section the main focus will be to explore one of the most used metaheuristics: Genetic Algorithms (GA). Additionally, some of the features added to GA to solve sequencing problems will also be explored.

Genetic Algorithms can be compared to the biological genetic evolution. In each generation, the biological code changes and as generations go by they evolve and, therefore, the principles of natural selection proposed and stated by Darwin for natural evolution also apply. Moreover, the concept of *survival of the fittest* is also applied to this algorithm. This concept is used to find an optimal or near-optimal solution.

There is no rigorous definition for genetic algorithms but the GA method is always applied with the same elements ([7]):

1. The solution provided can be said to be an *individual*
2. An *individual* belongs to a *population*
3. An *individual* is a set of *chromosomes*
4. A *chromosome* is a string of *genes*
5. *Alleles* are the values of the *genes*

GA repeatedly modifies a population of individual solutions. At each iteration a new population is created. Additionaly, the individuals of a new population are the *offspring* of the randomly selected *parents* of the previous generation.

The concept of the *survival of the fittest* is considered when selecting individuals for *mating*. Even though it is said to be a random process, GA often requires a *fitness* function that assigns a level of preferrence to each individual created and these new individuals are used to select the mating parents to produce the next generation [7].

After selecting the parents with better fitness, their chromosomes are combined - *crossover*. Local individual changes may also occur to garantee the populations' diversity - *mutation*.

The random key genetic algorithm, or RKGA, represents and encodes chromosomes with random numbers that range between 0 and 1. A decoder uses the strings or vectors of random numbers and associates them to a solution of the problem ([7]). Similarly to the GA, RKGA evolves the populations of chromosomes in each generation. So, after determining the fitness of each individual through the decoder, the population is divided in two groups: elite population and population of non-elite individuals.

This approach uses this elitist method to make sure that the elite population is inhereted unchanged to the next generation, making it a monotonically improving heuristic ([7]).

Like GA, RKGA uses mutation to insert variability into the populations and, therefore, avoiding getting caught in a local minimum.

In the RKGA approach, a mutant is also a string or vector of random numbers just like the ones created for the initial population.

For each generation, the mutation feature adds constantly mutant individuals. Therefore, each generation, iterated after the initial, will be composed by a mutant population, an elite population (chosen in the preceding generation) and a non-elite population generated by the process of mating randomly pairs of the previous generation (crossover).

RKGA was the study subject of [9]. It is stated that this algorithm was initially developed to solve complex scheduling problem in the automaker industry. For this type of problems RKGA proved to be efficient given different random seed values. Additionally, this algorithm found solutions whith less tardiness than those found by other methods with similar resources.

Based on the tests conducted using specific data sets, it is said that RKGA was able to quickly find good solutions and takes additional time to improve them, which makes this a good method to be applied in real scheduling implementations since the algorithm can be stopped sooner with almost no impact on its performance.

The authors then state that RKGA is a good algorithm to provide solutions for problems with equivalent complexity found in the automaker industry.

A biased random-key genetic algorithm (BRKGA) differs from the RKGA in the way the parents are selected for mating. In BRKGA, each individual from the next generation is the result of the mating of an elite and a non-elite parent of the previous generation. Additionally, a parent can be selected for mating more than once.

Mutation in the BRKGA is similar to what happens in the RKGA and occurs given a predetermined probability that can be defined in the framework.

BRKGA also allows to define the offsprings probability of inhereting part of the elite parent characteristics. This is called *parametrized uniform crossover*.

Figure 2.3 illustrates the parametrized uniform crossover for the BRKGA ([7]). Given a probability of inhereting an allele from the elite parent of 70%, and considering a randomly generated number with the same structure as the chromosomes. The offspring generated will assume the allele from the elite parent if the outcome of the random number is less than or equal to 0.7, otherwise it inherits the allele from the non-elite parent.

In [7] some application examples are given, for instance, communication networks, schedulling, manufacturing cell formation, two-dimensional packing and concave-cost network flow optimization.

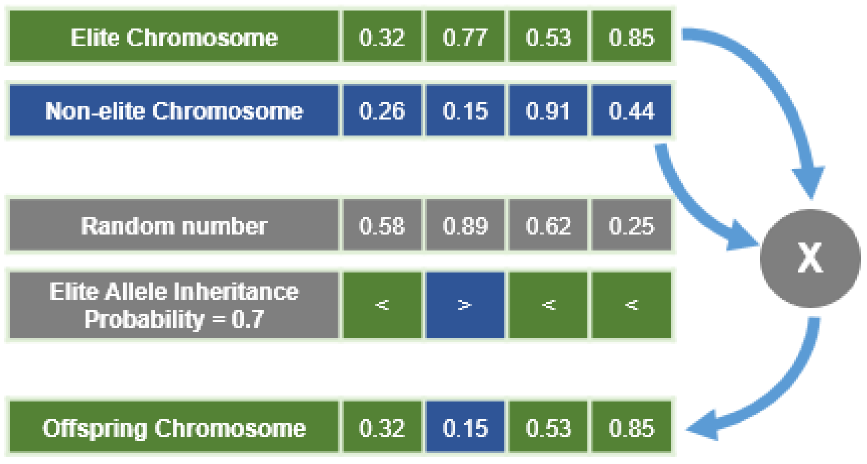


Figure 2.3: Parametrized uniform crossover

In [10] the author developed a mixed integer linear programming model to aid in the routing online system of grocery shopping. Although this is a growing market segment it stands a challenge due to its demanding logistics and marketing. One of the main problems is the in-store order picking, that consists in the fulfillment of customers' orders when performed in a supermarket or grocery store. This problem is directly associated with a large sum of the logistics' costs discussed before. The BRKGA was the approach chosen to solve this problem due to its elitistic evolution feature.

In this work the author compares the results given by the BRKGA method and the results of the Adaptative Large Neighborhood Search (ALNS) heuristic. For the comparison 10 problems were analysed using both methods. The author states that BRKGA was able to outperform in half of the tests conducted.

The study concludes that BRKGA presented high quality solutions with little computacional effort, thus being an efficient and effective metaheuristic able to aid online grocers to obtain more profit and a sustainable business model.

Motivated by the growing complexity of the production systems, [11] proposes a BRKGA approach for a hybrid flowshop scheduling problem. Similarly to [10] the authors had the objective of minimizing the total tardiness, i.e., the sum of all jobs tardiness.

In this case, the problem was represented by a mixed integer linear programming model and solved with BRKGA. This paper states that the tests were conducted in 432 large instances with different decoder versions of the BRKGA, then a comparison was made between them considering the literature reviewed. One of the versions was based on the bottleneck stage identification and, based on the results, the authors claim that this method stood out from the others considered, achieving better solutions in 61% of the instances.

Chapter 3

Solution Methodology

This chapter presents a solution methodology for the problem proposed by Follow Inspiration. As required, the suggested solution is focused on the scheduling of several tasks using multiple agents. Additionally, it's expected that the limited resources and the operations priority is also taken into consideration. This type of problem is commonly called a Resource Constrained Project Scheduling Problem (RCPSP).

Considering a shop-floor with multiple jobs, AGVs and workstations the challenge is to make all of it work autonomously and as efficiently and effectively as possible.

The already existing routing algorithm defines that each AGV is associated with a set of tasks consisting in a series of movements and this is called a mission.

BRKGA has proven its capability of solving this type of problems, thus being the chosen heuristic to solve the problem at hand. Moreover, in [7] the authors mention the experimental results using BRKGA in a RCPSP and, as stated, it outperformed others.

In this context, each possible solution represents different scheduling orders for the ten missions at hand. These missions are performed by ten AGVs as well so that each one of them has only one mission assigned. Furthermore, these so called missions consist in making the AGVs move from a starting point of the map to other three points to perform their activities. These movements simulate the work required by the AGVs in an industrial environmental where they need to exit a stop station, move to a warehouse interface to pick an item, deliver the item to a workstation and return to another stop or resting station while waiting for a new assignment.

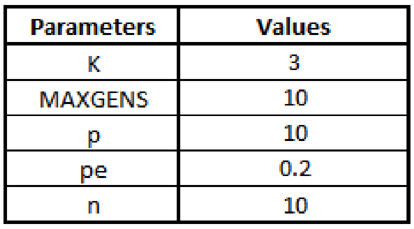
The BRKGA framework used in this study was adapted from [12] and it will be explained in this chapter. Nevertheless, [12] describes in great detail how the brkgaAPI works, its variables and the advantages of using such an efficient and easy-to-use application. For this study, it is important to highlight that this frameworks' structure is distinctively divided in two parts: *problem-independent* and *problem-dependent* part.

3.1 Problem-Independent

The problem-independent portion of the BRKGA doesn't have any information concerning the problem itself. This part of the metaheuristic processes everything related with the unchangeable components, i.e., everything that can be applicable or reutilized to any problem.

This module's encoding phase depends on a set of parameters to be defined in the framework, namely: the quantity of populations to consider (K), the maximum number of generations to evolve the populations (MAXGENS), the size of each population (p), the portion of elite parents (pe) and new random-key mutant elements (pm) in each generation, the number of genes in a chromosome (n) and the probability of an offspring inhereting an allele from an elite parent (ρ_e). The parameters' influence that was studied in this thesis were rhoe and pm because they are the main drivers of the metaheuristc's search throughout the solution space, since they're responsible for the converging speed and the search variability. All other parameters were fixed and their values are presented in Table 3.1.

Table 3.1: Parameters



Parameters	Values
K	3
MAXGENS	10
p	10
pe	0.2
n	10

The flowchar represented in Figure 3.1 shows how the BRKGA works for each generation.

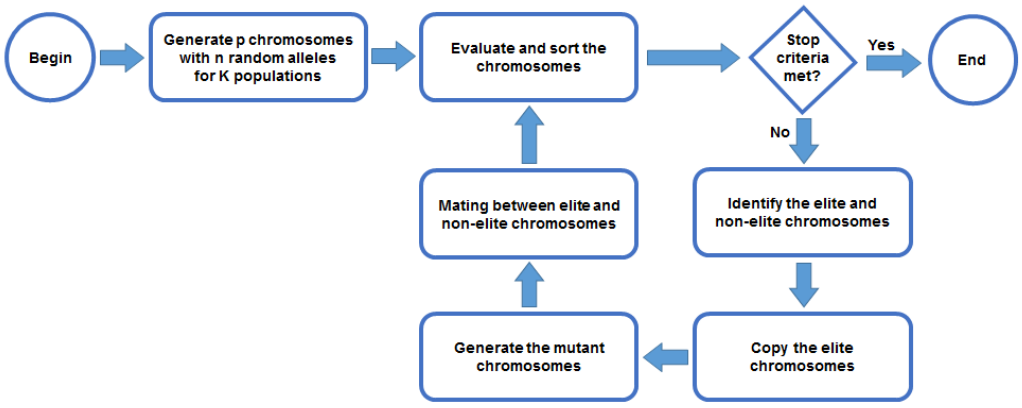

Begin
Generate p chromosomes with n random alleles for K populations
Evaluate and sort the chromosomes
Stop criteria met?
Yes
End
No
Identify the elite and non-elite chromosomes
Copy the elite chromosomes
Generate the mutant chromosomes
Mating between elite and non-elite chromosomes

Figure 3.1: BRKGA Flowchart

As seen in Figure 3.1, this algorithm follows 6 main steps:

1. Creates the first generation of chromosomes for K populations of random keys. The size of each population is p, and each chromosome has n alleles.

2. Sorts the solutions by their fitness value.

3. Classifies the solutions as elite or non-elite.

4. Copys the best elite solutions to the next generation based on their fitness value.

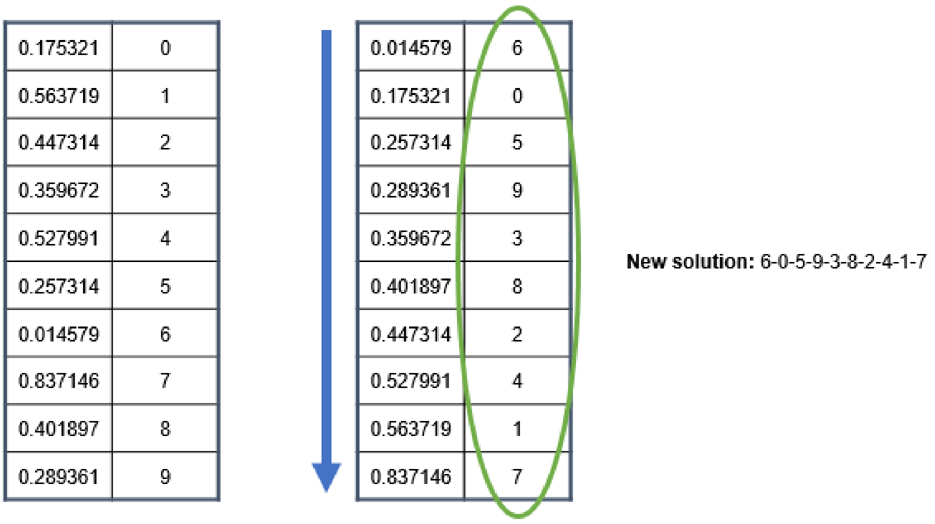
5. Generates the mutant population to be added to the next generation.

6. Combines non-elite and elite solution to create offspring by the mating process described in section 2.3. . Then, the cycle repeats from 2. until a stopping criteria is met, in this case a maximum number of generations was defined.

3.2 Problem-Dependent

The problem-dependent module of the BRKGA consists mainly in a decoder that produces solutions for the problem at hand from the random-keys produced by the problem-independent module. Additionally, the decoder also calculates the fitness values for each solution according to the users needs'.

In this case, each solution represented by a chromosome of n alleles is seen as a set of missions that will be reorganized in crescent order with respect to the gene value as seen in Figure 3.2. Moreover, the decoder also evaluates the solutions' fitness.



0.175321	0
0.563719	1
0.447314	2
0.359672	3
0.527991	4
0.257314	5
0.014579	6
0.837146	7
0.401897	8
0.289361	9

0.014579	6
0.175321	0
0.257314	5
0.289361	9
0.359672	3
0.401897	8
0.447314	2
0.527991	4
0.563719	1
0.837146	7

Figure 3.2: Decoding process of the random-keys

The objective function measures the quality of the solutions. In this specific problem there are two parameters considered, with different priority levels:

- MaxTime: instant of time when the last AGV stops

- NumStops: total number of stops the AGVs needed to make while waiting for the availability of a certain path

The objective function is calculated by giving a weigth of 2 and 1 for the MaxTime and NumStops respectively, as seen in equation 3.1. These variables were previously divided by a predetermined factor defined by Follow Inspiration, in order to normalize their values to vary between 0 and 1.

$$Fitness = \frac{2}{3}MaxTime + \frac{1}{3}NumStops \tag{3.1}$$

Since this is a minimization problem the lowest value is the best outcome.

Chapter 4

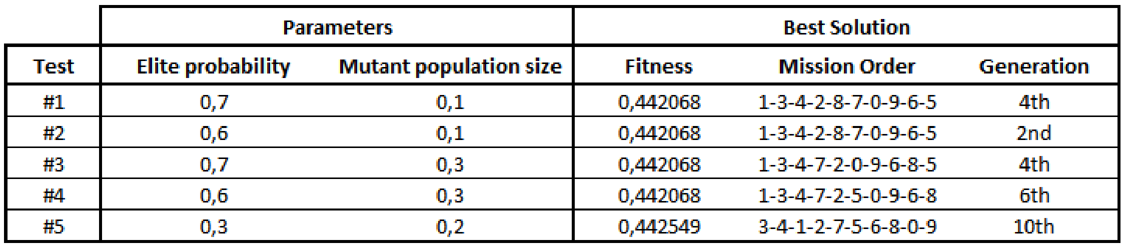
Results

In this chapter are presented the results of executing the algorithm with different values for two parameters:

- ρ_e : probability of inhereting an allele from the elite parent
- p_m : portion of mutant elements in the next generation

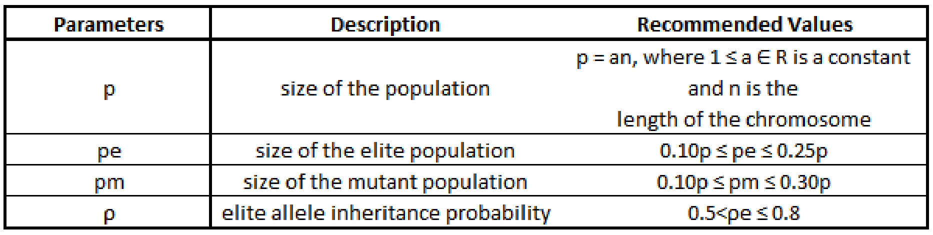
The results and changes made to the variables are presented in Table 4.1, and Table 4.2 shows the recommended settings defined by [7]. The run test that led to the best results can be consulted in Appendix A.

Table 4.1: Results and changes made throughout the testing phase



	Parameters		Best Solution		
Test	Elite probability	Mutant population size	Fitness	Mission Order	Generation
#1	0,7	0,1	0,442068	1-3-4-2-8-7-0-9-6-5	4th
#2	0,6	0,1	0,442068	1-3-4-2-8-7-0-9-6-5	2nd
#3	0,7	0,3	0,442068	1-3-4-7-2-0-9-6-8-5	4th
#4	0,6	0,3	0,442068	1-3-4-7-2-5-0-9-6-8	6th
#5	0,3	0,2	0,442549	3-4-1-2-7-5-6-8-0-9	10th

Table 4.2: Recommended parameter settings



Parameters	Description	Recommended Values
p	size of the population	p = an, where 1 ≤ a ∈ R is a constant and n is the length of the chromosome
pe	size of the elite population	0.10p ≤ pe ≤ 0.25p
pm	size of the mutant population	0.10p ≤ pm ≤ 0.30p
ρ	elite allele inheritance probability	0.5<pe ≤ 0.8

In the first test the parameters had the same values as the ones used by default in the framework. With this setup the local optimum was achieved at the 4th generation. Additionally, in the

7th generation a substancial number of solutions had the same fitness value as the best one so far, suggesting that the search is converging.

On the second test, considering the results obtained from the first one, the probability of inhereting an allele from the elite parent was lowered to spread the search through the solution space. Despite leading to the same convergence pattern, this test achieved the local optimum sooner than the first test (2nd generation).

For the third test, and to insert variability into the solution search space, the portion of mutant elements in the population was increased. The results achieved were similar to the ones obtained with the first test.

The fourth test consisted in changing both parameters at once and observe how the generations evolved. The local optimum was achieved at the 6th generation and in the 7th the results were converging to a local minimum with a fitness value of 0.442549. At the 10th generation the solutions tend to the optimal solution known so far.

One last test was conducted going against the elitist approach that characterizes the BRKGA. The result was the worst of all, the optimum was reached in the last generation, thus proving the advantages of the elitistic feature to achieve the best results faster.

Figure 4.1 shows graphically how the generations evolved and how fast they got to the optimum solution.

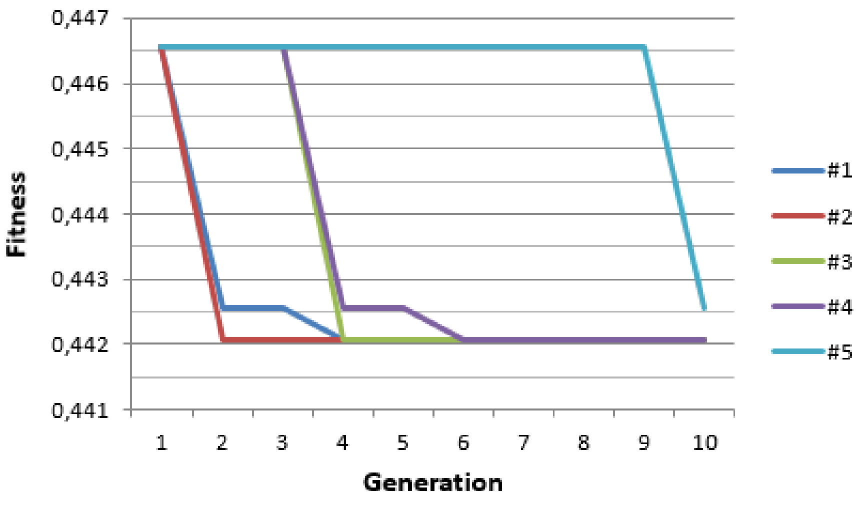


Figure 4.1: Results

By analysing the tests' results, both graphically and the generations' evolutions, it can be said that the test with the best results was the second one.

All the tests were conducted on an Intel Core i7-4720HQ CPU with 2.6 GHz and 8GB of RAM, running a Windows 10 64-bit based operation system, and using Microsoft's Office Visual Studio 2012.

Chapter 5

Conclusions and Future Work

This thesis' goal was to investigate and develop a new method for scheduling multiple tasks with multiple vehicles (AGVs), considering a set of requirements considered as a priority to Follow Inspiration S.A..

Since this was a follow-up of the already existing work developed by the company, the BRKGA had to be able to comunicate and use the given routing algorithm to schedule a set of missions.

The BRKGA was the approach chosen and, even though it can be improved, the results show that going further with this method can be beneficial and a starting point for a more sophisticated framework.

Considering the success in merging both routing and scheduling algorithms, and the results achieved, it is concluded that this investigation goal was accomplished.

In the future, the chromosomes' fitness evaluation algorithm could be improved in order to reduce its runtime and provide a faster approach to test several different solutions.

Appendix A

Test Example

Test 2

Generation	Max Time	Num Stops	Fitness
1	120,745	6	0,646557
1	127,357	7	0,704346
1	120,745	5	0,613224
1	120,745	4	0,579891
1	123,997	3	0,558586
1	123,997	4	0,591919
1	125,367	2	0,53032
1	123,997	4	0,591919
1	134,096	3	0,595937
1	133,629	3	0,594209
1	127,824	2	0,53941
1	119,66	4	0,575881
1	125,064	5	0,629198
1	129,729	6	0,679785
1	133,629	9	0,794207
1	120,745	2	0,513225
1	125,415	7	0,697164
1	119,66	2	0,509215
1	120,686	5	0,613007
1	134,121	6	0,696027
1	120,745	3	0,546558
1	130,359	4	0,615448
1	120,745	0	0,446559
1	123,997	9	0,758584
1	130,359	3	0,582115
1	119,53	3	0,542067
1	134,121	5	0,662694
1	127,824	0	0,472744
1	119,66	4	0,575881
1	125,367	8	0,730318
2	125,367	3	0,563653
2	130,423	0	0,482355
2	122,05	4	0,584719
2	123,997	3	0,558586
2	125,367	2	0,53032
2	123,997	3	0,558586
2	127,357	8	0,737679

Test 2

Generation	Max Time	Num Stops	Fitness
2	127,357	4	0,604347
2	120,745	0	0,446559
2	123,997	8	0,725251
2	119,66	2	0,509215
2	130,423	5	0,64902
2	119,66	7	0,67588
2	125,415	5	0,630498
2	119,53	4	0,5754
2	120,561	5	0,612547
2	119,53	0	0,442068
2	127,849	4	0,606167
2	120,745	0	0,446559
2	134,096	3	0,595937
2	127,824	0	0,472744
2	127,824	0	0,472744
2	127,357	7	0,704346
2	127,357	3	0,571014
3	133,629	8	0,760874
3	130,423	0	0,482355
3	129,729	3	0,579786
3	127,357	0	0,471015
3	130,423	0	0,482355
3	120,686	3	0,546341
3	130,423	2	0,549021
3	125,39	4	0,597074
3	120,745	2	0,513225
3	119,66	2	0,509215
3	130,359	3	0,582115
3	119,742	2	0,509518
3	119,66	2	0,509215
3	119,66	6	0,642547
3	120,745	2	0,513225
3	125,064	2	0,529199
3	120,745	0	0,446559
3	127,357	0	0,471015
3	120,686	4	0,579674
3	119,66	1	0,475882

Test 2

Generation	Max Time	Num Stops	Fitness
3	127,824	0	0,472744
3	127,824	1	0,506077
3	127,824	1	0,506077
3	130,542	4	0,616128
4	120,686	2	0,513008
4	130,359	4	0,615448
4	125,367	3	0,563653
4	119,53	1	0,475401
4	119,53	0	0,442068
4	125,367	3	0,563653
4	120,745	0	0,446559
4	120,745	0	0,446559
4	119,742	1	0,476185
4	130,359	0	0,482116
4	120,745	0	0,446559
4	119,66	2	0,509215
4	130,359	4	0,615448
4	119,742	2	0,509518
4	120,745	0	0,446559
4	120,745	5	0,613224
4	120,745	0	0,446559
4	119,53	0	0,442068
4	125,367	2	0,53032
4	119,742	2	0,509518
4	127,824	1	0,506077
4	134,096	3	0,595937
4	134,096	3	0,595937
4	120,686	3	0,546341
5	119,53	0	0,442068
5	125,367	4	0,596986
5	119,53	0	0,442068
5	130,359	4	0,615448
5	119,53	0	0,442068
5	124,918	2	0,528659
5	119,53	1	0,475401
5	128,986	3	0,577037
5	120,745	0	0,446559

Test 2

Generation	Max Time	Num Stops	Fitness
5	120,745	0	0,446559
5	119,66	0	0,442549
5	127,357	2	0,537681
5	127,357	1	0,504348
5	120,745	0	0,446559
5	120,745	0	0,446559
5	133,629	4	0,627542
5	120,745	0	0,446559
5	127,824	1	0,506077
5	119,53	2	0,508734
5	119,66	0	0,442549
5	127,357	1	0,504348
5	119,742	2	0,509518
5	119,742	3	0,542851
5	125,064	8	0,729197
6	125,367	2	0,53032
6	119,53	0	0,442068
6	120,745	2	0,513225
6	130,359	3	0,582115
6	119,53	0	0,442068
6	119,53	0	0,442068
6	125,367	2	0,53032
6	122,05	4	0,584719
6	127,357	3	0,571014
6	127,357	1	0,504348
6	119,66	0	0,442549
6	127,357	5	0,63768
6	120,745	0	0,446559
6	119,66	1	0,475882
6	122,474	0	0,452955
6	129,729	1	0,51312
6	119,66	0	0,442549
6	119,66	0	0,442549
6	125,367	4	0,596986
6	119,53	0	0,442068
6	119,53	0	0,442068
6	127,357	7	0,704346

Test 2

Generation	Max Time	Num Stops	Fitness
6	127,357	3	0,571014
6	127,357	1	0,504348
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	127,357	3	0,571014
7	130,359	5	0,648781
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,66	2	0,509215
7	130,359	0	0,482116
7	119,53	0	0,442068
7	122,66	8	0,720308
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	119,53	0	0,442068
7	124,032	5	0,625383
7	119,53	0	0,442068
7	119,66	0	0,442549
7	123,997	4	0,591919
8	119,53	0	0,442068
8	119,53	0	0,442068
8	119,53	0	0,442068
8	120,686	2	0,513008
8	120,686	4	0,579674
8	119,53	0	0,442068
8	119,53	0	0,442068
8	119,66	1	0,475882
8	120,745	0	0,446559
8	119,66	0	0,442549
8	134,298	5	0,663348

Test 2

Generation	Max Time	Num Stops	Fitness
8	119,53	0	0,442068
8	130,359	3	0,582115
8	119,53	0	0,442068
8	130,359	3	0,582115
8	120,686	3	0,546341
8	119,53	0	0,442068
8	123,997	5	0,625252
8	119,66	0	0,442549
8	119,53	0	0,442068
8	119,53	0	0,442068
8	120,686	3	0,546341
8	119,53	0	0,442068
8	133,629	4	0,627542
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	120,686	4	0,579674
9	119,53	0	0,442068
9	130,423	0	0,482355
9	130,359	3	0,582115
9	119,53	0	0,442068
9	130,359	3	0,582115
9	119,66	0	0,442549
9	130,359	0	0,482116
9	129,729	3	0,579786
9	119,53	0	0,442068
9	120,561	9	0,745879
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	119,53	0	0,442068
9	120,745	6	0,646557

Test 2

Generation	Max Time	Num Stops	Fitness
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	133,629	3	0,594209
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	130,423	0	0,482355
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	119,53	0	0,442068
10	124,17	8	0,725893

References

[1] António Eduardo Santos. Logística baseada em AGVs. 2013.

[2] Game Programming. URL: http://theory.stanford.edu/{~}amitp/GameProgramming/.

[3] Joana Santos and Pedro Costa. Time enhanced A *: Towards the development of a new approach for Multi-Robot Coordination. 2015. doi:10.1109/ICIT.2015.7125589.

[4] Andry Maykol Pinto, F Rocha, A Paulo Moreira, and Paulo G Costa. Shop Floor Scheduling In a Mobile Robotic Environment. *Epia 2011*, pages 377–391, 2011. doi:10.1007/978-3-642-24769-9{_}28.

[5] António Paulo Moreira, Paulo J.Costa, and Pedro Costa. Real-time path planning using a modified A* algorithm. *9th Conference on Mobile Robots and Competitions*, (fig 2):141–146, 2009.

[6] Ibrahim H. Osman and James P. Kelly. *Meta-Heuristics: Theory and Applications*. 2012. URL: https://books.google.pt/books?hl=pt-PT{&}lr={&}id=e3TjBwAAQBAJ{&}oi=fnd{&}pg=PA1{&}dq=metaheuristics+and+heuristics{&}ots=E0EsgcrCtB{&}sig=uTLewfBahM1GDdwYqw3EDkRovDk{&}redir{_}esc=y{#}v=onepage{&}q=meta-heuristics{&}f=false.

[7] José Fernando Gonçalves and Mauricio G. C. Resende. Biased random-key genetic algorithms forcombinatorial optimization. *Journal of Heuristics*, 17(5):487–525, 2010. URL: http://link.springer.com/10.1007/s10732-010-9143-1, doi:10.1007/s10732-010-9143-1.

[8] Fred Glover and Gary A. Kochenberger, editors. *Handbook of Metaheuristic*.

[9] Bryan A Norman and James C Bean. A Genetic Algorithm Methodology for Complex Scheduling Problems. *Naval Research Logistics*, 46(January):199–210, 1998. doi:10.1002/(SICI)1520-6750(199903)46:2<199::AID-NAV5>3.0.CO;2-L.

[10] Tiago Miguel das Neves Salgado Ferreira. *In-store order picking routing: A Biased Random-Key Genetic Algorithm Approach*. PhD thesis, Faculdade de Economia da Universidade do Porto, 2015.

[11] Guilherme Barroso Mainieri. *BRKGA meta-heuristic for a scheduling problem in hybrid flowshops*. PhD thesis, 2014.

[12] Rodrigo Franco Toso and Mauricio G. C. Resende. A C++ application programming interface for biased random-key genetic algorithms. *Optimization Methods and Software*, 30(1):1–15, 2014. URL: http://www.tandfonline.com/doi/abs/10.1080/10556788.2014.890197, doi:10.1080/10556788.2014.890197.