UNIVERSITÉ DE MONTRÉAL

AUTOMATIC DISASSEMBLY TASK SEQUENCE PLANNING OF AIRCRAFTS AT THEIR END-OF-LIFE

ARASH AFSHARZADEH

DÉPARTEMENT DE GÉNIE MÉCANIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION

DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES

(GÉNIE MÉCANIQUE)

JUILLET 2016

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

AUTOMATIC DISASSEMBLY TASK SEQUENCE PLANNING OF AIRCRAFTS AT THEIR END-OF-LIFE

présenté par : AFSHARZADEH Arash

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. BALAZINSKI Marek, Docteur ès Sciences, président

M. MASCLE Christian, Doctorat, membre et directeur de recherche

M. BAPTISTE Pierre, Doctorat, membre et codirecteur de recherche

M. BARON Luc, Ph. D., membre

DEDICATION

To my dear Mom & Dad,

To my dear brother and sisters,

To my dear Nooshin

ACKNOWLEDGEMENTS

I would like to thank my director of research, Dr. Christian Mascle for accepting me in this project that provided me an opportunity to learn new interesting subjects and I am grateful of all of his supports and for all financial arrangements for this project.

I am also thankful of Dr. Pierre Baptiste for all of his time and interest in this work and for technically helping me in this project.

I also would like to thank Dr. Marek Balazinski for partially funding my research.

I would like to thank Dr. Baron Luc for taking the time to study this thesis and for his keen interest, questions and comments on this project.

I am thankful of all of the staff in mechanical engineering department for their help during my studies and my special thanks goes to Madams Martine, Carole and Myriam.

Finally, I would like to thank “Consortium de Recherche et d’innovation en Aérospatiale au Québec” (CRIAQ) and also industrial partners of CRIAQ ENV-412 project.

RÉSUMÉ

Une prise de conscience des problèmes environnementaux à l'échelle mondiale ainsi que des avantages économiques a stimulé les chercheurs à trouver les possibilités de réutiliser et de recycler les produits en fin de vie. Chaque année plusieurs centaines d'avions atteignent globalement fin de leur navigabilité et doivent être retirés du service actif. De ce fait, une attention accrue est maintenant accordée à la fin de vie des avions.

Désassemblage joue un rôle important dans la prise de décision de fin de vie. La faisabilité économique du processus de démontage avec beaucoup d'incertitudes est une préoccupation majeure limitant sa mise en œuvre dans la pratique de l'industrie. De nombreuses recherches dans le domaine de la planification et des opérations de processus de démontage a été fait, qui visent de plus en plus la faisabilité économique du démontage avec la réduction des temps de démontage de proposer des séquences de démontage optimisées. Par conséquent, ces dernières années, de nombreux chercheurs ont publié des articles sur la planification de la séquence de démontage des produits en fin de vie qui est un problème NP-complet optimisation combinatoire. Néanmoins, il y a eu un peu d'attention à la planification de la séquence de démontage d'avions en fin de vie.

Cette thèse aborde la planification de séquence de démontage des pièces réutilisables d'avions en fin de vie avant le démantèlement pour le recyclage. Puisque les composants récupérés vont être utilisés à nouveau, une approche non-destructive tout en respectant les instructions fournies dans le manuel d'entretien d'avion intitulé « Aircraft Maintenance Manuel » (AMM) pour le retrait des pièces est prise en considération.

Ordonnancement de désassemblage dans cette recherche ne traite pas le séquençage le démontage des pièces comme dans d'autres études, mais il planifie séquence de tâches de démontage dans l'AMM. Une tâche de démontage consiste combinaison d'opérations pour la préparation du démontage ou le procède de démontage pour un ou plusieurs pièces.

Tout d'abord, un modèle de séquençage de démontage est proposé par l'examen structure des tâches de démontage dans l'AMM. Ensuite, un code Matlab est développé qui lit la base de données énuméré des tâches et sous-tâches qui sont acquises à partir de l'AMM et génère la séquence de démontage des tâches et sous-tâches automatiquement en utilisant le modèle proposé. Le code est capable de générer des séquences de désassemblage de tâches pour n’importe quelle pièce sollicitée.

Enfin, un algorithme glouton et un algorithme glouton adaptatif sont évalués pour optimiser la séquence de démontage des tâches afin de minimiser nombre de changements dans les zones d'opérations de démontage. Les résultats générés dans le code Matlab, suggère l'efficacité de l'algorithme glouton adaptatif proposé.

ABSTRACT

An awareness of the world's environmental problems plus economic benefits has stimulated researchers to seek the opportunities to reuse and recycle end-of-life (EOL) products. Each year hundreds of aircraft globally reach end of their airworthiness and should be withdrawn from active service. Due to this fact, increased attention is now being paid to EOL of aircrafts.

Disassembly plays an important role in EOL decision making. The economic feasibility of the disassembly process with lots of uncertainties is a main concern limiting its implementation in industry practice. Many researches in the field of disassembly process planning and operations has been done that aim increasing economic feasibility of disassembly with reducing disassembly times with proposing optimized disassembly sequences. Consequently, in recent years, many scholars have published articles on disassembly sequence planning of EOL products that is a NP-complete combinatorial optimization problem. Nevertheless, there has been a scant attention towards disassembly sequence planning of EOL aircrafts.

This thesis addresses disassembly sequence planning of reusable components of EOL aircrafts before dismantling it for recycling. Since retrieved components are going to be used again, a non-destructive approach with respecting all instructions provided in aircraft maintenance manual (AMM) for removal of parts is taken into consideration.

Disassembly scheduling in this work does not deal with scheduling disassembly of components as in other works but it schedules sequence of removal Tasks in AMM. A removal task consists combination of operations for preparation of disassembly or process of disassembly for a part or multiple parts.

At first, a disassembly sequencing model with considering structure of disassembly tasks in AMM is proposed. Afterwards a Matlab code is developed which reads from enumerated database of tasks and subtasks that are acquired from AMM and generates disassembly sequence of tasks and subtasks automatically using the proposed model. The code is capable of generating disassembly sequences of tasks for any given removal task of solicited part.

Finally, a greedy and an adaptive greedy algorithm are proposed to optimize disassembly sequence of tasks with minimizing changes in visited zones of disassembly operations. Results generated in Matlab code, suggests effectiveness of proposed adaptive greedy algorithm.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

2D	Two-Dimensional
3D	Three-Dimensional
AFRA	Aircraft Fleet Recycling Association
AMM	Aircraft Maintenance Manual
AMTOSS	Aircraft Maintenance Task Oriented Support System
ASGAs	Algorithm of Self-Guided Ants
ASPEN	Assembly Sequence Planning and Evaluation System
ATA	Air Transport Association
BOM	Bill Of Material
CAD	Computer Aid Design
CO	Close Out
CRIAQ	Consortium de Recherche et d'Innovation en Aérospatiale au Québec
DFD	Design For Disassembly
DFIG	Disassembly Feasibility Information Graph
DPN	Disassembly Petri Net
DPP	Disassembly Process Plans
DSP	Disassembly Sequence Planning
EASA	European Aviation Safety Agency
EOL	End Of Life

FDOM	Fuzzy Disassembly Optimization Model
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
HBN	Hybrid Bayesian Network
JSU	Job Set-Up
JSUI	Job Set-Up Information
MTM	Methods Time Measurement
NP	Non-deterministic Polynomial-time
OEM	Original Equipment Manufacturer
P	Procedure
PAMELA	Process For Advanced Management Of End-Of-Life Aircraft
PN	Perti Net
PPX	Precedence Preservative Crossover
PSO	Particle Swarm Optimization
RFID	Radio-Frequency Identification
TS	Topological Sort

CHAPTER 1 INTRODUCTION

1.1 Problem statement

Technically life of an airplane is close to indefinite. As long as aircraft is serviced and there are spare parts available, an aircraft can be kept airworthy. The business life of an aircraft, however, is limited. Maintenance costs will increase when an aircraft becomes older and bear in mind that new aircraft with newer technology has higher passenger comfort and lower utilization cost, such as fuel and maintenance cost. When an aircraft has reached the end of its business life, it will be retired and withdrawn from service. Over the next 20 years, it is estimated that approximately 12000 airplanes, which are currently in service for different purposes, will reach their End of Life (EOL) (Asmatulu et al., 2013). Consequently, this question raises that what we can do with hundreds of aircrafts arriving at the end of life. Stock them in deserts or crush them to recycle materials. There are several alternatives, but original equipment manufacturers (OEMs) are also attentive that any alternatives, as a global analysis, should consider different criteria of sustainable development. Upon reaching the end of aircraft airworthiness, the airline/owner decides what to with the aircraft. From environment and economic point of view abandonment or unlimited duration stock of airplanes are not wise choices. Therefore, the owner may decide to perform a reconversion activity (commercial to freight for example) or sell the retired aircraft for being disassembled (The process comprising all the activities required to remove all the valuable components from an aircraft, which can be re-used in another aircraft), dismantled (The process comprising all the activities required to make it possible to recycle materials from an aircraft) and subjected to a recovery of its materials for recycling.

As it was explained, we understand that it is a significant need to recycle materials or re-use parts and equipment of dismissed aircrafts that are still in a good state of functioning. In this context, the Project CRIAQ ENV412 was defined to allow university researchers to study on sustainable development and end of life of aircrafts. Current thesis will only discuss retrieving of selective and non-destructive disassembly of high value parts, which are considered reusable.

For the re-use/resale of retrieved items and new assemblies as previously used parts, the disassembly of aircraft parts is subject to following best practices (AFRA, 2013). Furthermore,

reusability of removed parts and equipment is conditional upon meeting all required maintenance procedures. The procedure of disassembling parts is quite same as if the aircrafts parts were in phase of maintenance, thus the operators should respect procedures in Aircraft Maintenance Manual (AMM).

In this study, we do not have access to design data, 2D or 3D drawing of parts or general assemblies of airplane. Therefore we do not have distances between parts or precedence relationships among them. Also we do not have a database that provides costs of disassembly operations, the tools utilized and man-hour estimates of activities. The only document which is handed over to us is the AMM in the PDF format files which a computer code reads these files as texts and retrieves enumerated tasks. In the AMM, duration of disassembly tasks is not provided but order of execution of procedures for each of tasks is written in there. Each of tasks in AMM has a task number and is enumerated in accordance with ATA standard.

Since an airplane is a large product, during disassembly of reusable parts at EOL stage, different areas of disassembly operations will be existed. Therefore, disassembly tasks are dispersed in different zones of airplane. In some cases of part removals, platforms, technicians with specified skills and training or special equipment is needed. The disassembly operations are labour intensive and disassembly project manager needs to plan for different teams of technicians and acquisition of equipment. With considering a lean approach, which eliminates waste from process, we do not like to change our working zone as much as possible. Therefore, we are interested in an approach to minimize changing working zones and displacements of labor and equipment consequently.

The target is generating an optimized disassembly sequencing plan for retrieving reusable parts of an aircraft its EOL. The optimization factor is total number of zones changed in a disassembly project.

The AMM is more than 3000 pages with more than 45000 tasks. This requires power of a computer software to aid automatic data acquisition from AMM and to generate disassembly task sequencing and applying optimization algorithms. For this reason a Matlab code is developed.

1.2 Literature review

1.2.1 Disassembly sequencing

Disassembly for recovery is defined as the systematic extraction of valuable parts of an outdated product with goal of re-using or recycling extracted parts. Disassembly as a process was performed in the slaughterhouses more than a century ago. However, the "systematic approach" of disassembly has relatively a short history of 30 years.

Disassembly is an important process in retrieving some value of outdated products since it allows for the selective retract of desired parts. In the literature, because of this benefit many studies has been done in the area of disassembly.

In other hand, many papers focus on disassembly for the purposes of maintenance and repair, particularly if these had to be performed in environments such as space stations, nuclear power plants, etc. with harsh or hard to access environments. In these cases, the design should foresee and allow for automated or remote disassembly and reassembly.

Lambert (2003) categorizes a variety of purposes of works on disassembly sequences as following:

- Remote construction and repair in inaccessible or hazardous environments such as in spacecraft and nuclear equipment.
- Optimal repair and maintenance.
- A tool for assembly optimization.
- Design and optimization of disassembly lines.
- Optimum product design regarding the product's end-of-life phase, which is called Design for Disassembly (DFD).

Disassembly sequencing is about the problem of finding the best order of disassembly tasks in the retrieving of different parts of a product. Disassembling of complex products requires mathematical modelling and a connection representation aimed at selecting a good or optimum sequence of disassembly tasks. To find the best or optimum path, in most cases heuristics, metaheuristics or mathematical programming is applied on the model.

In the literature on disassembly sequencing, two basic approaches can be recognised: (a) the mechanical approach that is mainly used in the analysis of mechanical assemblies, and (b) the hierarchical tree approach that is mainly applied to electrical and electronic equipment. The mechanical approach originates from the assembly study, and the hierarchical tree approach is related to the Bill of Materials, which is applied in Materials Resource Planning.

Disassembly is considered to be as the reverse process of assembly method by many disassembly researchers and then, most authors in disassembly area have researched in assembly area. Takeyama et al. (1983) said, "*The sequence of assembly is the reverse of that of disassembly*".

Lambert (2003) summarized essential differences between assembly and disassembly as following:

- Disassembly is usually not performed to its full extent: incomplete disassembly is often preferred, which adds the disassembly depth to the decision variables.
- The assembly process is often not completely reversible (decisions need to be made between destructive and non-destructive disassembly).
- The value added in disassembly processes is usually modest compared to that obtained in assembly.
- Uncertainty exists with regard to the quality of the components,
- Uncertainty exists in the supply of discarded products from both qualitative and quantitative points of view,
- In disassembly, a variety in supplied products might be present.
- Due to mentioned features, disassembly is mainly carried out by human labour instead of by automated assembly lines and robots.

Bourjault (1984) did the first attempt and formulated assembly sequencing problem using a systematic approach in his doctoral thesis. He presented automatic generation of assembly sequences of a product with using rules by answering a series of "yes" or "no" questions about mating of parts for an assembly and modeled a product by utilizing the information contained in a BOM and an assembly drawing to form a liaison graph, where the components are the nodes and

the liaisons are the arcs representing the mates. An Algorithm determined assembly sequences using the liaison graphs.

De Fazio and Whitney (1987) proposed simplified set of rules, “precedence constraints”, by asking two questions about liaison precedence. Questions ask about “what liaisons must be done prior to doing liaison i” and “what liaisons must be left to be done after doing liaison i”.

Khosla and Mattikali (1989) developed a methodology that uses software to automatically generate assembly sequences from a 3D model of the assembly.

Baldwin et al. (1991) also utilized CAD for automatic assembly sequence generation.

Kanai et al. (1996) developed a computer aided Assembly Sequence Planning and Evaluation system (ASPEN) that takes all the solid model components of a product and automatically determines all feasible sequences by decomposition. ASPEN also determines the optimum sequence using Methods Time Measurement (MTM) as time standards for operating time determination.

Zhang and Kuo (1997) proposed a graph-based approach to find feasible disassembly sequences from the CAD system directly and automatically.

Gungor and Gupta (1997) developed a methodology to evaluate different disassembly strategies to choose the best among them. They also proposed a disassembly sequence generation heuristic, which gives a near optimum disassembly sequence.

Moore et al. (1998) proposed an algorithm which automatically generates a disassembly Petri net (DPN) from a geometrically-based precedence matrix. They analyzed the resulting DPN to generate all feasible disassembly process plans (DPPs), and cost functions were utilized to determine the optimal DPP.

O'Shea et al. (1998) provided the state of the art different issues particular related to disassembly planning. The subjects covered issues from product representation, to task analysis, task representation, sequencing, clustering, life cycle engineering issues, and cost estimation. Materials data requirement issues and some practical applications in disassembly plant.

Gungor and Gupta (1998) discussed the uncertainties and its sources in disassembly sequence planning (DSP) and presented a methodology to deal with uncertainty in DSP implementation for products with defective parts.

Srinivasan and Gadh (2000) analysed the problem of removing one or more components of an assembly, defined as selective disassembly, and presented geometric abstractions and representations for automated selective disassembly analysis of geometric models.

Hsin-Hao et al. (2000) developed the economic analysis method of the disassembly process and develop an artificial neural network approach for disassembly sequence generation problem. Due to combinatorial nature of disassembly sequencing problem, there is an increasing trend in the use of metaheuristics.

Lazzerini and Marcelloni (2000) presented a Genetic Algorithm (GA) for generating optimal assembly sequences. Population generation randomly generated usually non-feasible sequences, however they defined crossover and mutation operators to allow the GA to evolve to feasible optimal sequences. The algorithm were applied to sequences consisted of up to 17 components.

Kuo (2000) addressed the disassembly sequence and cost analysis for the electromechanical products during the design stage. He divided disassembly planning into four stages: geometric assembly representation, cut-vertex search analysis, disassembly precedence matrix analysis, and disassembly sequences and plan generation. The disassembly cost is categorized into three types: target disassembly, full disassembly, and optimal disassembly.

Seo et al. (2001) presented a genetic algorithms based heuristic approach for an optimal disassembly sequence considering economic and environmental aspects.

Murayama et al. (2001) presented a methodology of generating disassembly sequences for component replacement at maintenance stages using information entropy evaluation and heuristics for efficient generation of disassembly sequences. They evaluated disassembly based on two criteria: disassembly time and disassemblability.

Erdos et al. (2001) investigated the modelling and evaluating product end-of-life options that is the problem of representing products and finding disassembly sequences with the goal of maximizing revenue. They developed algorithms to generate the product recovery graph, to find optimal disassembly plans that maximize revenue using the generated product recovery graph with uncertainties of the end-of-life products.

Gungor and Gupta (2001) presented a systematic approach to generate an optimum disassembly sequence plan for product recovery. They generated a disassembly precedence relationships matrix

representing the geometric precedence information using CAD representation of the product. Afterwards, they utilized a branch-and-bound heuristic to find an optimum disassembly sequence.

Tang et al. (2002) surveyed the state of art in disassembly modelling, planning and its application.

Rai et al. (2002) presented a disassembly sequence generation methodology using the Petri net technique by correlating cost indices combined with heuristic search procedures. The presented heuristic generates and searches a partial reachability graph to arrive at an optimal or near-optimal disassembly sequence based on the firing sequence of transitions of the Petri net model.

Lambert (2003) surveyed the literature on disassembly sequencing and summarised and categorised work of more than 200 articles in the area of disassembly sequencing.

Torres et al. (2003) presented an algorithm based on a representation method for products, which shows the hierarchical relationships among components of the product. They established an algorithm for partial non-destructive disassembly sequence of a product. The disassembly sequence were obtained automatically from design CAD data.

Mascle and Balasoiu (2003) addressed a wave propagation based disassembly algorithm to select the disassembly sequence of a specific component of a product.

Lambert (2005) presented the search for optimum disassembly sequences for products that can be represented by disassembly precedence graphs and that are subjected to sequence-dependent disassembly costs. He applied an exact approach, based on iteratively solving a binary linear programming.

Li et al. (2005) presented an object-oriented intelligent disassembly sequence planner for maintenance based on the disassembly constraint graph using genetic algorithms to generate near optimal disassembly sequence from all the feasible combination of the disassembly operations.

Lambert and Gupta (2005) in their book reviewed disassembly related issues and presented comprehensive details about methods, approaches, and applications in this field.

Dong et al. (2006) presented an approach to automatic generation of disassembly sequence from hierarchical attributed liaison graph of an assembly through recursively decomposing the assembly into subassemblies. Their approach integrates general geometric reasoning with the knowledge about how to generate feasible and practical disassembly sequences.

Chung and Peng (2006) presented an evolutionary algorithms approach to generate a feasible and optimal plan for selective disassembly in remanufacturing, ensuring both batch disassembly of components and tool accessibility to fasteners.

Kongar and Gupta (2006) addressed a genetic algorithm for disassembly sequencing of EOL products.

Sarin et al. (2006) developed a disassembly optimization problem as precedence constrained asymmetric traveling salesman problem and developed an iterative procedure to minimize the costs associated with the disassembly process while maximizing the benefits resulting from the recovery of components and subassemblies that constitute the product.

Kang and Xirouchakis (2006) surveyed the disassembly sequencing problems in which second-hand or EOL products are disassembled into basic parts and components, especially due to repair and maintenance.

Langella (2007) presented heuristic algorithm with considering demand for disassembled part for remanufacturing of products.

Shimizu et al. (2007) applied genetic algorithms to address a system supporting strategic decision-making on disassembly for recycling at the design stage of the product life cycle. The issue of uncertainty modelling and management arises in the context of the optimal disassembly planning problem, one of the problems to be addressed by remanufacturing processes.

Reveliotis (2007) addressed a reinforcement learning approach for providing optimal solutions to the optimal disassembly problem considering uncertainties.

Kim et al. (2007) provided a literature review on works done in disassembly scheduling area.

Giudice and Fargione (2007) presented a genetic algorithm approach to disassembly process planning that supports the search for the disassembly sequence best suited for both aspects, service of the product and recovery at the end of its useful life.

Dong et al. (2007) offered an approach based on accessibility and EOL strategy, using Petri net modelling to generate an optimal disassembly sequence. Their approach utilize AND/OR graphs to generate all feasible disassembly sequences, and then graphs are transferred into Petri net graphs while accessibility values and life span values of components are taken into consideration to obtain the optimal disassembly sequence.

Duta et al. (2008) developed an evolutionary algorithms approach for the multi-criteria optimization problem of the disassembly scheduling.

Lambert and Gupta (2008) presented a heuristic algorithm for disassembly sequencing problems subjected to sequence dependent disassembly costs using disassembly precedence graph of a cell phone with twenty-five parts.

Wang et al. (2008) presented a GA method to solve disassembly sequence planning problem. To describe product disassembly sequence and operation information, they presented disassembly feasibility information graph (DFIG). Then, disassembly sequence planning problem is mapped onto the DFIG as an optimal path-searching problem. In GA a chromosome represents the sequence of disassembly operations, and they applied the precedence preservative crossover (PPX) as crossover operator for establishing next generation of chromosomes. The generated chromosomes which does not respect precedence relations are rejected.

Adenso-Díaz et al. (2008) claimed since determining an optimal disassembly sequence planning is NP-complete and complex as well as a challenging problem to solve, heuristic and metaheuristic techniques may be the only practical ways to solve such problems. They considered two criteria to look for: First, a sequence that its cost is close to their cost aspiration. Second, a sequence that prioritizes some selected parts to be disassembled as early as possible. In their article they presented a greedy randomized adaptive search procedure (GRASP) and path-relinking-based heuristic methodology to solve this bi-criteria disassembly planning problem. The proposed algorithm were applied to four different products with a maximum of 100 components in each.

Tripathi et al. (2009) proposed a metaheuristic named Algorithm of Self-Guided Ants (ASGAs) to solve the computationally hard fuzzy disassembly optimization model (FDOM) problem. Their algorithm aims to maximize the net revenue generated from disassembled components from an EOL product. The factors considered for their model are number of parts, number of subassemblies, number of joints, setup costs, joint breaking costs, recovery value and disposal costs.

Grochowski and Tang (2009) presented a machine learning approach based on a DPN and a hybrid Bayesian network (HBN) to model the disassembly process and predict the outcome of each disassembly action by examining the probabilistic relationships between the different aspects of the disassembly process.

Puente et al. (2010) presented an algorithm to sequence tasks to be performed in the disassembly of a specific material from a product or to separate all the materials of a product. They utilized hierarchical model of a product with having disassembly time of components.

Yun and Moon (2011) proposed a GA approach with utilizing a topological sort (TS) based procedure for solving precedence constrained sequencing problems. The TS-based procedure used in the proposed GA approach can generate feasible sequences in precedence constrained sequencing problems. The proposed algorithm is applied to a problem with a maximum of seventy-five nodes.

ElSayed et al. (2011) developed an automated disassembly cell for online (real time) partial disassembly operations of PCs at their EOL. Their cell consists a robotic arm, a camera, range-sensing and component segmentation visual algorithms. The authors applied an online GA approach for finding near/optimal disassembly sequencing. The precedence preservative crossover (PPX) methodology was utilized for the crossover operation. The fitness value was obtained by the overall disassembly time.

Go et al. (2011) reviewed several disassembly methods to enhance the recovery of EOL products, especially for vehicles, so that valuable parts are efficiently retrieved
Liu et al. (2012) applied an improved max–min ant system based algorithm to generate a near-optimal solution for the problem of product disassembly sequence planning in a virtual maintenance environment with considering all disassembly constraint information, such as adjacency and constraint relations between parts and subassemblies, disassemble time, disassemble tool, disassemble priority, quality, reliability and load of disassemble machine, etc

Smith et al. (2012) introduced an approach for creating, and searching methods of disassembly sequence structure graph model for multiple-target selective disassembly sequence planning. The approach uses expert rules to choose parts, part order, and part disassembly directions, based on contact, motion, and fastener constraints.

Go et al. (2012) utilized a GA approach implemented in Matlab in order to determine the optimal disassembly sequence of an engine of an EOL vehicle. The applied method achieves the minimum disassembly time with considering precedence relationships.

Wan and Gonnuru (2013) proposed the use of radio-frequency identification (RFID) technology to support disassembly decisions for end-of-life products. They presented a fuzzy-based disassembly

planning and a disassembly sequencing model to maximize net profit and utilized GA approach for finding an optimal solution.

Tian et al. (2013) presented chance constrained programming models based on the uncertainty feature of a disassembly process. In their model removal times, labour costs and disassembly direction changes is taken into consideration. Authors proposed two hybrid intelligent algorithms to solve the proposed models: one integrating stochastic simulation and neural network, and another integrating stochastic simulation, genetic algorithm and neural network.

Popescu and Iacob (2013) proposed a disassembly method based on two elements: The connection interface concept and the mobility operator that integrates information on geometrical constraints, contact surface relative position and common area, neighboring components, and relative mobilities of the assembly components from the 3D CAD model of a product. Heuristic rules and unit ball concept for determining component mobility is utilized.

Wang et al. (2013) reviewed the fundamental methodology and its development of computerized intelligent disassembly planning research. It is concluded that heuristic algorithms are increasingly being used to search for the optimal solution to improve efficiency. However due to complexity of the matter for more accurate results there is a demand to integrate intelligent methods such as genetics algorithm or artificial networks. They also argue that disassembly planning is essentially a process that needs person's knowledge and experience of engineering, so solely a perfect mathematical optimization theory is not a way to the final resolution.
Zhao et al. (2014) presented a reasoning algorithm using a Fuzzy reasoning PN for optimisation of a disassembly process based on the maximal reclaiming profit of retrieved parts.

Rickli and Cameliob (2014) investigated impact of EOL product quality variability, represented as an EOL product age distribution, on the optimal or near-optimal partial disassembly sequence. Moreover, they utilized a GA heuristic based on sequence feasibility, expected profit standard deviation and profit probability per EOL product.

Xia et al. (2014) proposed a simplified Teaching–Learning-Based Optimization algorithm for solving disassembly sequence planning problems. The algorithm utilizes population-based evolutional method and consists of three parts: a Feasible Solution Generator to generate a feasible disassembly sequence, a Teaching Phase Operator and a Learning Phase Operator used to learn

and evolve the solutions towards better ones by applying crossover operation that preserves precedence relationships.

Giri and Kanthababua (2015) proposed a method to generate the disassembly sequences for mechanical products, which uses the part interference matrix that contains the removal directions of the parts and the part connection graph that indicates the contact among the parts in the assembly.

For generating the part connection graph, 2D views generated from the CAD assembly model for automatically identifying the part removal directions is utilized. They applied a heuristic method to generate the best feasible disassembly sequences.

Issaoui et al. (2015) presented a solution space reduction for disassembly sequencing problems. Connection tree of a target component is constructed with using its CAD model. The generation of sequences is based on reading of connection tree branches and elimination of infeasible ones. They applied their method to a product with 22 components.

Riggs et al. (2015) presented a two-stage optimization algorithm that first determines the optimal partial disassembly sequence according to reuse value of components, and in next stage adds sequence dependence task times and finds the optimal partial disassembly sequences.

Berg et al. (2015) presented an application of immersive computing technologies to aid in evaluating disassembly sequences in early design. The application displays both 3D geometry of a product and an interactive graph visualization of existing disassembly sequences. For presented work, authors generated the disassembly graphs manually. The calculated optimal path can be highlighted allowing the user to compare the optimal sequence against alternatives.

Mitrouchev et al. (2015) presented a method based on the lowest levels of a disassembly product graph instead of considering the geometric constraints for each pair of components. It is discussed that if the product has a modular design, this will allow aggregating some parts of the assembly in modules and reaching the target without dismantling the components one by one. The proposed method first generates disassembly geometry contacting graph of the product and then generates the feasible disassembly sequences. Moreover, authors proposed concept of micro-units that eliminates all the components that are unrelated to the target ones, permitting reducing the number of iterations for disassembly sequence generation and consequently reducing the search time.

Wang et al. (2016) introduced a method for evaluation of disassembly operations in Virtual Environment. A set of five criteria is proposed: visibility of the sub-assembly part, disassembly angles, stability of the subassembly, number of tools' changes and path direction change. The presented method can be used for disassembly sequence generation either at early stage of design or at EOL.

Alshibli et al. (2016) presented a metaheuristic algorithm for disassembly sequencing, with using a sensory system and an online real-time Tabu search applied to a robotic arm that executes disassembly operations. The objectives of the proposed algorithm are to minimize the traveled distance by the robotic arm, the number of disassembly method changes, and the number of robotic arm travels by combining the identical-material components together. The authors state that the applied Tabu search runs faster than GA approach for the similar problem.

1.2.2 Aircraft at its end-of-life

Until a few years ago, EOL aircrafts were abandoned. Starting in the 2000s, the two largest aircraft manufacturers Airbus and Boeing began to get interested in the management of aircrafts to develop alternative approaches of how to handle aircrafts at their EOL. Airbus launched the PAMELA project (Process for Advanced Management of End-of-Life Aircraft), as an aircraft dismantling demonstration project with support from the European Commission's "LIFE" initiative under the classification of "waste management, recycling and reduction of landfill".

The main concept was implementing strategies to develop a set of best practices, and standards, for the dismantling of an aircraft safely and with respect for the environment.

During the PAMELA project (Airbus, 2008), the consortium created a three-stage process approach of handling EOL aircraft, called 3D approach. Table 1-1: Table 1-1 shows summary of this 3D approach.

Table 1-1: PAMELA 3D approach

D1- Decommissioning	- Cleaning & Decontamination - Draining of tanks - Implementation of safety procedures under EASA (European Aviation Safety Agency) Part 145
D2- Disassembly	- Equipment removal under EASA Part 145 - Parts removal
D3- Smart Dismantling	- Final draining of systems - Dedicated removal of material - Aircraft Deconstruction & Categorization of materials - Shipping of extracted categorized materials to waste treatment channels

In other hand, in 2006 eleven charter companies established AFRA (Aircraft Fleet Recycling Association) with the goal to organize and present an industry perspective on aircraft sustainability through the development and recommendation of best practices and technologies for the management of the world's older fleet.

The AFRA has published "Best Management Practice for Management of Used Aircraft Parts and Assemblies and for Recycling of Aircraft Materials" (BMP). BMP is a document that represents a collection of recommendations concerning best practices for the management of parts that are removed from an aircraft at its EOL.

- The disassembly: intended to guide aspects of the management of parts removed from an EOL aircraft.
- The recycling: intended to guide aspects of the processing of aerospace materials in order to transform these into usable materials.

Law does not enforce AFRA's procedures and guidelines. However, AFRA provides accreditation to companies in the field of dismantling and disassembling aircraft parts. It gives confidence to clients who want to use the spare parts, which are retrieved from EOL aircrafts that they will use the parts that are in good state and all the best practices has been considered at the time of retrieval from airplanes.

Another important study in this context is a major research project named CRIAQ ENV-412 that is an initiative of the "Consortium de recherche et d'innovation en aérospatiale au Québec" (CRIAQ) and funded by Natural Sciences and Engineering Research Council (NSERC) and partners Bombardier Aerospace, Bell Helicopter Textron Inc., Aluminerie Alouette, BFI Canada, Sotrem-Maltech, CRIAQ, Nano Québec and Mitacs with collaboration of École Polytechnique de Montréal, Centre technologique en aérospatiale (CTA), Université Laval, École de technologie supérieure (ETS) and McGill University.

1.2.3 Disassembly sequence planning for aircrafts

Even though the literature is abound with disassembly sequencing, except the works done in context of CRIAQ-ENV412 project, one cannot find studies specifically dealing with issue of disassembly sequence planning of aircraft re-usable part at EOL of aircraft.

There is just a very few studies concerning disassembly of parts, which is for the purpose of maintenance of aircraft parts.

Cheung et al. (2005) and Cho et al. (2009) discussed staff rostering to facilitate the allocation of labor resources at the time of aircraft maintenances.

Christiand and Yoon (2007) proposed a GA approach for determining an optimal assembly/disassembly path planning for maintenance of aircraft parts. Constraints such as obstacles and the part distances was taken into consideration.

Zhang et al. (2008) presented a component disassembly approach for aircraft assembly process based on fuzzy-clustering algorithm.

Hassan and Yoon, (2010) presented a disassembly path planning algorithm for maintenance of aircraft parts considering geometry constraints with the purpose of optimizing time of disassembly.

Zhong et al. (2011) proposed metaheuristic methods for maintenance of large equipment parts including aircrafts. They presented a Dijkstra's algorithm and particle swarm optimization (PSO) to determine optimal disassembly sequences. The proposed method was applied to disassembly sequences for maintenance of a nose landing gear system of a regional jet with ten components to be disassembled. The authors state that their proposed method reduce the calculation complexity, and it is faster than a GA approach.

1.2.4 Conclusion of literature review

As presented in previous section study of disassembly sequencing has started about thirty years ago and has attracted many scholars. However, apparently researchers are not done with this issue due to its complexities. Many improvements has taken place but limits are still there. Each disassembly case and its constraints is different with another one and a solution that best fits that specified problem should be determined.

Table 1-2 summarizes disassembly sequencing problems into three categories: applications, representations and methodologies. As it can be seen from methodology section, there are quite considerable number of techniques that are used for determination of optimal disassembly sequences. Nevertheless, seems that publication with utilizing genetic algorithm approach are more than other approaches. However, this method also has its own limits and that is why scholars are still seeking other methods.

The state-of-the-art about the disassembly sequence planning of aircrafts shows that there is just a couple of studies with the application of maintenance of aircrafts and there is a big gap in literature in this subject. Actually, the only related work towards disassembly sequence planning of EOL aircraft parts has been initiated in CRIAQ-ENV412 and Camelot (2012) has addressed this subject in her master studies.

Current work presented in this thesis was defined as continue of Camelot's study with the purpose of checking and validation of her model and algorithm, and to automatically generate disassembly sequences in case of verification/modification of proposed model. Therefore, Camelot's approach is investigated thoroughly and details will be presented in next chapters.

Table 1-2: Disassembly sequence planning

Application	Design For Disassembly Maintenance End-Of-Life Disassembly
Representation	Disassembly Precedence Graph Disassembly Tree State Diagrams And/Or Graphs
Methodology	Graphical, Linear Programming, Mixed Integer Linear Programming, Dynamic Linear Programming, Heuristics, Metaheuristics, Ant Colony Optimization, Artificial Immune System, Artificial Neural Network, Bayesian Network, Differential Evolution, Evolution Algorithm, Evolution Programming, Fuzzy System, Genetic Algorithm, Greedy Randomized Adaptive Search Procedure, Memetic Algorithm, Multi-Agent System, Neighborhood Search, Particle Swarm Optimization, Path Relinking, Reinforcement Learning, Scatter Search, Simulated Annealing, Tabu Search, Multi-Criteria Analysis, Stability Analysis, Precedence Relations, Modularity Analysis, Parallelism, Uncertainty, Case-Based Planner, Economic Analysis, Environmental Cases, 2D Views, 3D Models, Mechanical Assemblies

1.3 Objective

In this study, target is to have an optimized disassembly sequence of reusable parts of an aircraft in its end-of-life, leading to an automatic generation of a disassembly process with respecting maintenance procedures.

In literature review, many different approaches for optimizing a disassembly sequencing were explained. Obviously when talking about optimizing, there should be a parameter to optimize. The main parameters which was consider in researches is reducing time and costs. For this most of studies offered had considered times of disassembly tasks for all operations or costs of any disassembly action or equipment utilized and with applying their proposed optimization algorithm could find an optimized disassembly sequence.

However, for our study, we do not have any of such databases. The only resource that is available is the Aircraft Maintenance Manual. Hopefully since tasks in AMM is standardized by ATA for all type of airplane, (no matter who the OEM is), we have access to ATA chapter of is each task which is part of task's name in the AMM. Consequently, we can understand the task is performed in which zone of the aircraft.

Since an airplane is a large product, during disassembly or reusable parts at EOL stage, different areas of work will be existed. Therefore, disassembly tasks are dispersed in different zones. In some cases of part removals, platforms, technicians with specified skills and training or special equipment is needed. To plan for different team of technicians and acquisition of equipment, with considering lean approach, which eliminates waste from process, we do not like to change our working zone frequently. Therefore, we are interested in an approach to minimize changing working zones and displacements of labor and equipment consequently. With considering this concept, disassembly operations has to be generated by maximizing the number of executed tasks in a working zone. The procedure has to address all necessary tasks and subtasks requested by AMM to remove parts.

AMM has more than 45000 tasks. That is why generating a disassembly sequencing procedure needs is not something than can be easily done manually and that is why a Matlab code is utilized for this purpose.

1.4 Thesis structure

In next chapter, an overview of methodology and approach of current work is presented. In Chapter 3, discussions about a model for disassembly sequencing of aircraft parts is provided. After reviewing principles in section 3.1, previous model of disassembly sequencing is investigated in section 3.2 and problems with that model is outlined. Since the previously proposed model is not applicable, a correct model is developed which is described section 3.3.

Afterwards, automatic generation of disassembly tasks with using computer aid and validation of the proposed model is discussed in section 3.4.

Determination of an optimized disassembly task sequences and utilized algorithms and its efficiencies is discussed in Chapter 4.

Contributions and recommendations is summarized in Chapter 5.

CHAPTER 2 METHODOLOGY

2.1 Studying of previous works

As discussed earlier this project is a successive to work of Camelot (2012). Camelot in her research did a study on available data and methods to be applied on a disassembly sequencing of an EOL aircraft. Since the study is for the parts or systems that are re-usable, the methodology should only study non-destructive methods of parts retrievals. Consequently, all the disassembly operation should respect the AMM. Eventually Camelot came up with a model of disassembly and presented an imaginary disassembly graph and developed an algorithm for finding a disassembly sequence.

In another project, Yongliang Cai, developed a computer code for the purpose of data acquisition from the AMM manual. In this work, Cai changed the format of PDF files of AMM to TEXT files and with using a Matlab code, found all of the tasks and subtasks in each AMM file (Mascle et al., 2014). Output of this code can be used to find subtasks of a task automatically via searching the generated database.

The purpose of current project is to integrate the model to database for automatic generation of an optimized disassembly sequence. For this reason, a very detailed study was required on those pervious works, first to understand their work and later utilize the concepts.

2.2 Developing new model

With studying the previous existing model, I concluded that, the model violates the AMM requirements that need respecting precedence constraints. Therefore, in this project the author developed correct model. Problems of the previous model, and new model will be presented in next sections.

2.3 Validation of the model

The main input the process is the AMM files. Model should respect the AMM procedures. With integrating the model to automatically read tasks from manual, there is only a manual approach to see if the generated sequences respects the AMM.

With doing this manual verifications, using many different random input tasks, author observed that the output has a problem and is not in accord in manual.

With different root cause analysis, I could figure out that the problem was becoming from the code that read from AMM. As it was the input to the model, it affected the output.

Different problems were found in the output of the code guidelines and errors feedbacks were made to developer of the code. With this collaborating and by many different random analysis on output of code, the problem was solved.

Consequently, input to the model was corrected and correction in output of the model was observed.

2.4 Finding an optimized disassembly sequence

In literature for the purpose of generating an optimal or near optimal disassembly sequence parameters such as cost of each disassembly action, time of each disassembly action, number of tool changes or revenue generated from disassembled parts has been elaborated.

As it was discussed earlier, for this work, we do not have any data regarding previously mentioned parameters. Nevertheless, with using concept ATA chapter, we can figure out in which zone of aircraft the disassembly operation is performed. As a matter of fact with eliminating unnecessary change of work zones, time and cost of disassembly can be improved. In this manner, project manager can plan for special task force with skills and tools required in a specified zone of the aircraft. Thus, we are interested in an approach to minimize changing working zones and displacements of labor and equipment. With considering this concept, disassembly operations has to be generated by maximizing the number of executed tasks in a working zone.

2.5 Automatic generation of sequences of disassembly

At the time of disassembly when we are visiting a zone, the disassembly model provides other available tasks that can be performed in that zone. An algorithm finds this zone and when all the available task in that zone are preformed, the algorithm finds the next best zone.

A Matlab code is developed, which can perform all of these procedure and with giving removal task number of required parts, it automatically generates a model and sequences of disassembly tasks.

CHAPTER 3 MODEL FOR DISASSEMBLY OF RE-USABLE AIRCRAFT PARTS

3.1 Principles

3.1.1 TASK numbering in AMM

As discussed earlier, we have the AMM as our input source and our desire is to eliminate excessive displacement with minimizing visiting different zones.

In the AMM, for each of actions such as servicing, removal of parts or system and installing them a task number has been assigned. This task number has the format of XX-XX-XX-XXX-XXX where X is a number. For example, a task number in manual will look like 32-10-12-400-801. Fortunately, no matter who the OEM is, ATA has standardized the first three elements which each consist two digits (XX-XX-XX) in ATA spec 100 standard and later in ATA iSpec 2000 standard with some modifications in ATA spec 100. These standards contains format and guidelines for technical manuals written by aviation manufacturers and suppliers and is used by airlines and other segments of the industry in the maintenance of their respective products. These documents provide standard for aircraft systems numbering, often referred to as ATA system or chapter numbers.

First element of a task number is its ATA chapter that refers to a system, the second element is its subsystem (section) and the third element is the unit (subject). Together, these elements identify the applicable aircraft hardware.

The XXX-XXX at the end of the task number is called AMTOSS index (Aircraft Maintenance Task Oriented Support System). The first three digit of AMTOSS, which is the fourth element of a task number, defines the maintenance function being performed. The fifth element of a task number is a three-digit number used to create unique numbers for all tasks or subtasks which are similarly numbered through the first four elements. Tasks are numbered from 801 through 999. Subtasks are numbered from 001 through 800.

At the time of disassembly of aircraft, there are tasks that have the same first three elements (XX-XX-XX). Therefore, one can compare these numbers and verifies that if the disassembly is taken place in a same zone. The definition of zone can go to fewer depths if a user like to choose just two elements (XX-XX). This will lead to wider areas in zones.

Figure 3-1, demonstrates an example of a maintenance task. With considering ATA standardization, we can assign all of tasks that have 32-10-12 at its beginning to a same zone.

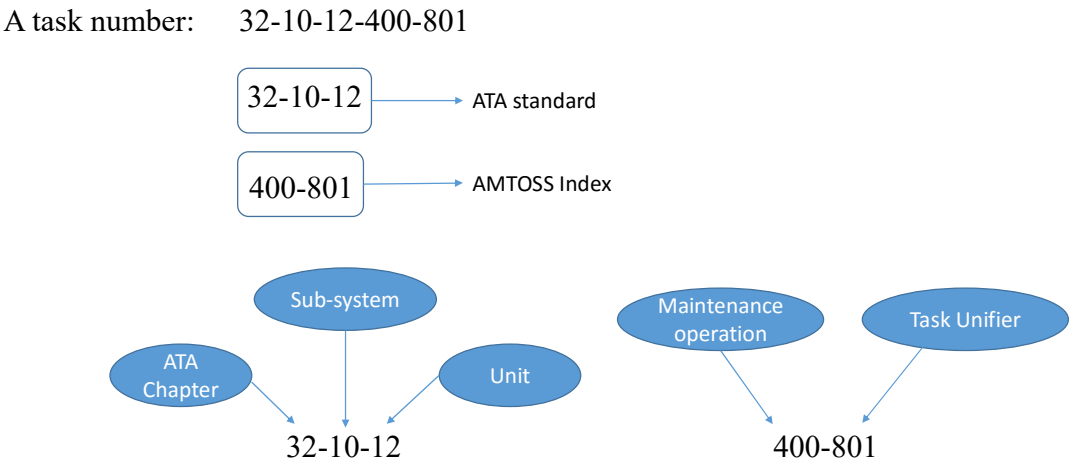


Figure 3-1: An example of maintenance task in AMM

3.1.2 AMM structure

AMM contains all of the tasks that address disassembly, assembly, service, inspection, etc. of an aircraft. The subjects in the maintenance manual are divided into following categories:

- Description and Operation
- Maintenance Practices
- Servicing
- Deactivation/Reactivation
- Removal/Installation
- Adjustment/Test
- Inspection/Check

- Cleaning/Painting
- Repairs

As discussed in previous section, all of these tasks have a task number that is a unique number for that specified task. A task consists all of the procedure that should be performed on an aircraft to satisfy the needed operation. Each task is structured by subtasks and can also refer to other tasks in the AMM.

A subtask is a logical division of a task, and is used to divide tasks into understandable parts. A subtask generally contains only one type of information such as a table, a reference to a task, or a simple procedure. For each subtask same as tasks, a number is assigned but subtasks are not referenced by tasks or other subtasks.

Each of task of AMM has for main sections:

- Job Set-Up Information
- Job Set-Up
- Procedure
- Close-Out

Job Set-Up Information (JSUI): This section of the task consists all of the information and list of tasks and tools that is required to perform the task. Therefore, in this section an operator can have an overview of the work that should be done.
Job set-up (JSU): This section of the task lists and describe all of the preparation operations and set-ups before performing the main process.
Process (P): This section is for the subtasks and tasks that the main operations are taking place.
Close-out (Co): This section is for the subtasks that is performed to return the state of the aircraft to the state which was before performing Job set-up and Process tasks.

It is a remarkable note in AMM that we should pay attention in our disassembly-sequencing problem:
"The steps written within the procedures for the various tasks or subtasks in the AMM are in logical order as interpreted by the Technical Writer. They are correct if followed step by step for the completion of the task or subtask. However, if the intent of the maintenance procedure is not altered and the objective is met, the order of the steps required to complete the task or subtask may be

revised at the discretion of the technician (Example: step 4.f. before step 4.e.). Unless otherwise specified, the elapsed time between the start and the end of a procedure is not important."

This statement provides a guideline for us:

In our problem, we should consider the precedence logic that is provided in the AMM. Therefor at the time disassembly, tasks and subtasks should be executed in accordance with to order of appearance in the AMM. Figure 3-2 demonstrates a structure of a task in AMM and direction that should be taken for the executing tasks and subtasks.

Furthermore as it was said in previous chapter, the AMM does not provide elapsed times of procedures.

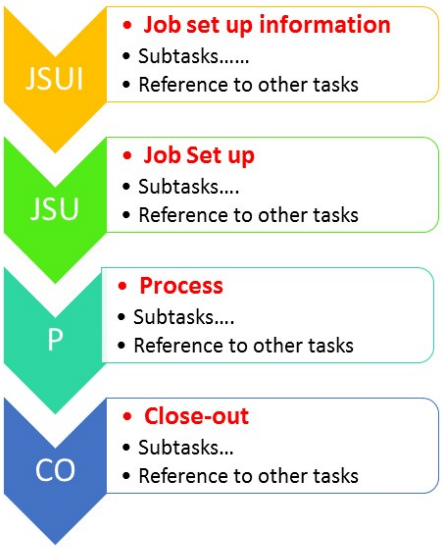


Figure 3-2: Structure of a TASK in AMM

3.1.2.1 Implementation of a task

In AMM, a task is divided to several subtasks. In each of subtasks maybe a procedure is described or maybe along with that procedure it can refer to the other tasks. All of these tasks and subtasks should be implemented in accordance with order of appearance in the AMM.

For example lets say we want to implement an imaginary task named: TASK 21-22-23-400-800. Therefore if with referring to AMM it is observed that there are fourteen tasks and subtasks, named T1, T2, …., T14, Figure 3-3 shows the sequence of implementing tasks.

Figure 3-3: Sequence of implementing tasks in AMM

It should be noted that it is possible that each of Tasks or subtasks of T1 to T14 refer to other tasks in AMM. In this case, again same principles applies.

Actually, there are several tasks that are needed to be performed for different tasks and those tasks are not uniquely preformed for a single operation. That is why technical writer of AMM assigns a task name to such kind of tasks and instead of repeating the procedure in text of several tasks, the technical writer just simply refers to name of that task.

At the time of operation, if the operator notices a referral to other task, he or she should find the procedure of implementing that task in AMM.

For example of Figure 3-3, any tasks or subtasks of T1 to T14 can refer to other tasks. In this case again same principle of Figure 3-3 applies for those referral tasks and operator should implement all of subtasks and tasks mentioned in AMM for any of T1 to T14 if any. Let us imagine that T4, T8 and T9 refer to other tasks. Figure 3-4: An example of a task with Figure 3-4 demonstrates such this case.

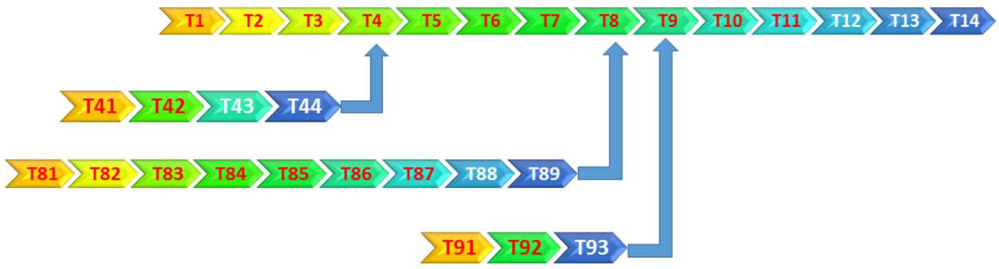


Figure 3-4: An example of a task with referral to other tasks

For above example operator performs T1, T2, T3 and reaches T4. He or she notices that in T4 there is reference to other tasks. Therefore, he/she should implement those tasks before moving to task T5. After tasks T41, T42, T43 and T44 are implemented then T5 should be performed.

After performing T7, its T8's turn to be executed. It is noticed that there are nine tasks which this tasks or subtask of T8 refers to. Again before being able to implement T9, the operator should execute tasks of T81, T82,…to, T89. The same goes for T9.

The referral to other tasks does not stop here! T41 to T44, T81 to T89 and T91 to T93, are tasks themselves. Any of these tasks can refer to other tasks. Therefore, same principle of previous paragraph applies. For example if T43 refers to other tasks, the operator should implement those tasks before being able to move to T44.

3.1.2.2 Tasks needed from AMM

For our purpose that is an automatic disassembly sequencing, two sections of a Task is required:

- Job set-up (JSU) section
- and process (P) section

Job setup information (JSUI) section is not required because it provides an overview of tasks in next sections. This information is redundant for our purpose and tasks are again repeated in next sections.

It is also important to note that, for our disassembly procedure we do not need to perform tasks in *close-out (CO)* sections. AMM states that this section is for returning the status of airplane to what has been before performing maintenance tasks. In fact, when we are disassembling parts at stage of end-of-life of aircraft, this issue is not our concern anymore.

3.2 Disassembly model in the previous work

3.2.1 Methodology

Ms. Camelot in her thesis and in (Camelot et al., 2013), presented a model for disassembly graph of an EOL aircraft. It supposed to be used in this project for generating automatic disassembly sequences with utilizing the output of code that Mr. Cai had developed, and provides digital database of all of tasks in AMM with their subtask. For this purpose, the author did a very detailed study of the existing model to be utilized. However, it was concluded that Camelot's model violates AMM procedures. Even though the model tries to respect maintenance procedure implied by the

AMM, but it contains fatal errors. In next paragraph, discussion is provided about fundamental of these errors.

Figure 3-5 demonstrates an imaginary model for a disassembly graph developed by Camelot. Each of tasks from t_1 to t_{19} should be executed and as discussed earlier each of these tasks can have tasks/subtasks. In this model, all of these subtasks of a task are aggregated in one node for each task. To represent this aggregation, on node of each task a fraction with two numbers is provided: The numerator is for status corresponding to the number of executed tasks/subtasks in that node. Obviously, at start of disassembly all of numerators in all nodes are zero. The denominator is total number of tasks/subtasks for that task. For example, t_{15} has five, t_{17} has seven, and t_{18} has 12 subtasks/tasks.

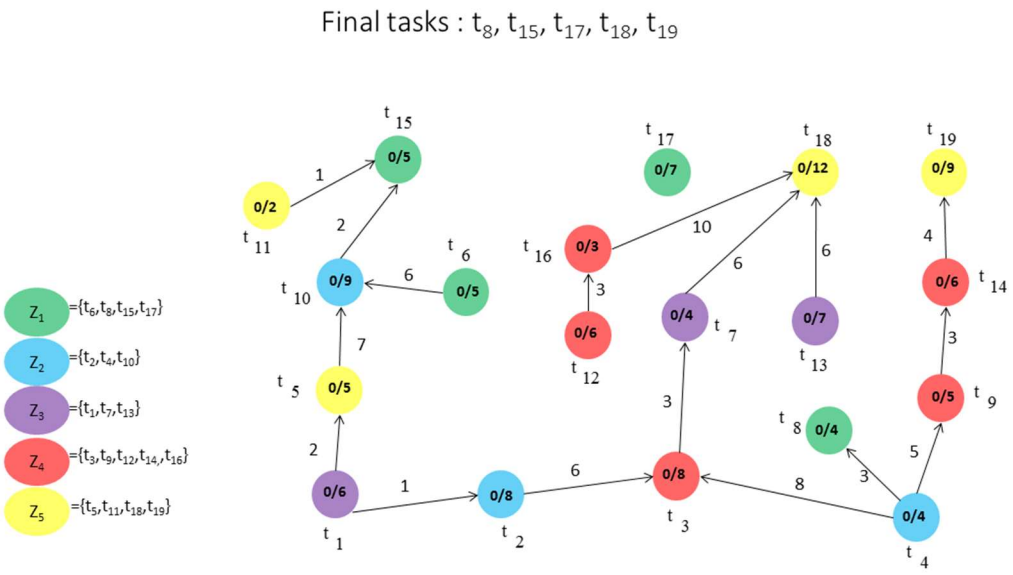


Figure 3-5: An example of disassembly graph model developed by Camelot

Edges represents a link between two nodes (tasks). It is used when one of tasks/subtask of a task calls another task. As shown in Figure 3-6, the number on the edge (m) represents the order of task/subtask in entering edge of node (task i) that calls the task in origin of edge (task j).

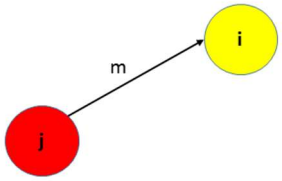


Figure 3-6: Edge between two tasks

For example in Figure 3-5, for task t_{18}, its 10^{th} task/subtask calls the task t_{16} and its sixth task/subtask calls the tasks t_7 and t_{13}.

Furthermore, in accordance with three double digit of task number which was discussed before (ATA 2200), for each of nodes/tasks, the zone which that task is executed is colored. In this representation, nodes with same color are those tasks that are in the same zone.

There are other details about this model such as mathematical representation, and how it is used to find an optimal disassembly sequence, which can be found in details in Camelot's publications. However, provided information to this extend would be enough to explain the problems with the model.

3.2.2 Problems with Camelot's model:

3.2.2.1 Aggregating subtasks in one node

3.2.2.1.1 Eliminating information about tasks/subtasks

Aggregating subtasks/tasks of a task in one node deletes all the attributes of those subtasks/tasks. In a less important matter, the disassembly model will not provide details of those tasks in that node. For example in Figure 3-5, at the time of disassembly operator is advised that t_{17} should be executed. This model just tells the operator that t_{17} has seven subtasks and does not provide any more details for which task should be implemented.

3.2.2.1.2 Wrong assignment of zones

Further to previously mentioned problem, a zone of disassembly is defined by the aggregated task number. However, a task can have many subtasks and with a quick review of AMM, it can be seen that, hardly ever those subtasks of a task are always in the same zone of the task.

For our purpose zone of operation, should be found by tasks inside of each node and there is highly probable that subtask belong to different zones. Therefore, aggregation of subtasks in their parent node will lead to fatal error in zone definition where operations are executed.

3.2.2.2 Not considering precedence relation between tasks/subtasks

As discussed in in 3.1.2.1 section an illustrated in Figure 3-4, any subtask of a task can refer to other tasks. The order of execution of those tasks is a logical order that the technical writer has provided and those tasks should be executed in accordance with order appearance in the manual.

For example in Figure 3-5 task t_{18}, 10^{th} task calls t_{16} and sixth task calls t_7 and t_{13}. As it is clear from the graph there is not any link between t_7 and t_{13}. These two subtasks are both called by sixth task of task t_{18} and there should be an order of execution for it in the AMM. As it has been for the t18 itself. So the model is violates itself an AMM.

Ms. Camelot in her thesis presented an actual graph for an actual task of CRJ100. That graph is shown in Figure 3-7. From this graph, all of previously mentioned problems can be seen again.

The main task has eight subtasks and it has been colored as a zone. Subtasks T1 to T7, have subtasks/task themselves with quantity of their denominators, however, a color that means all of these subtasks are in the same zone is assigned to these nodes.

Furthermore, it can be seen that the T1 to T7, all are called by fifth subtask of main task. However, any link between them has not been established that leads to an arbitrary order of execution of these tasks at time of disassembly. As discussed earlier, this is contrary to the logical ordering of tasks that technical writers of AMM manual have stated in AMM.

3.2.3 Conclusion

With this investigation of previous work and model, it is concluded that we do not have a viable model and the author needs to develop a new model.

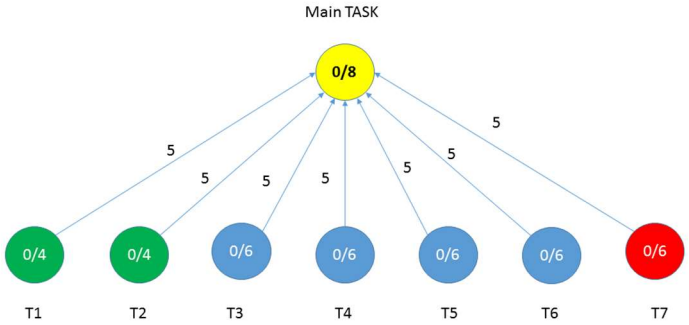


Figure 3-7 Camelot's disassembly graph for an actual task of CRJ 100/200 AMM

3.3 Developing a model for disassembly sequencing

3.3.1 Establishing a graph for disassembly of one part

If a disassembly 'Task' has four subtasks, the graph would be a directed graph that would look like a chain. Figure 3-8: Example of a Task with four subtasks. Figure 3-8 is an example of such this graph. This representation means for executing the 'Task', first subtask 1 (sub1), then subtask 2 (sub 2), then subtask 3 (sub 3) and finally subtask 4 (sub 4) should be executed.

In other hand, these subtasks can refer to (call) other tasks. All those tasks should be performed one by one, and once all those tasks in a subtask are executed, we can consider that specified subtask is completely done and we can move to next subtask for execution.

Tasks called by a subtask again should be executed in specified order in AMM; therefore, the precedency constraint should be constructed in the graph. Figure 3-9, shows order of execution of tasks for an imaginary example, if subtask 1 calls one task (T11), Subtask 2 calls three tasks of (T21, T22 and T23), Subtask 3 calls T31 and finally subtask 4 calls two tasks of (T41 and T42).

Our endeavor does not stop here. Any of tasks T11, T21, T22, T23, T31, T41 and T42 have their own subtasks and again those subtasks can call other tasks. This process should continue to the point that all the tasks which calls are made to them are found and listed in their corresponding subtask.

Figure 3-8: Example of a Task with four subtasks

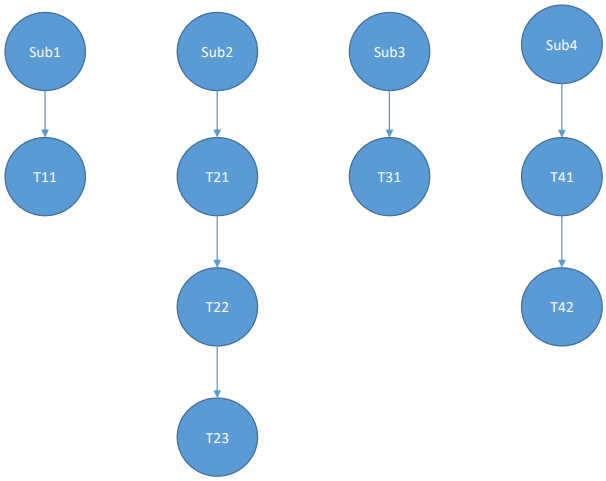


Figure 3-9: Subtasks referring to other tasks

For our example, for the simplicity we assume that T11, T21, T22, T23, T31, T41 and T42 do not call other tasks. Therefor at this time, a final graph can be generated. In Figure 3-10, this graph is demonstrated.

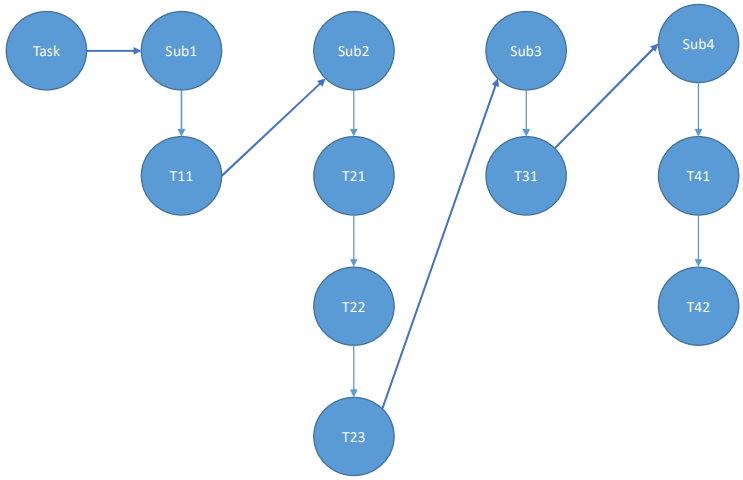


Figure 3-10: Graph of tasks for implementing ‘Task’

With retrieving this information from AMM, a sequence of execution of tasks and subtasks is generated, so for implementing ‘Task’ here is the sequences:

1. Execute Subtask 1
2. Execute T11
3. Execute Subtask 2
4. Execute T21
5. Execute T22
6. Execute T23
7. Execute Subtask 3
8. Execute T31
9. Execute Subtask 4
10. Execute T41
11. Execute T42

A computer code using Matlab is developed that generates these sequences automatically with getting the ‘Task’ as an input to the code. The methodology will be presented in the next sections.

3.3.2 Establishing a graph for disassembly of multiple parts

When we are dealing with an EOL aircraft there are several reusable parts that are going to be disassembled. For each of these parts there is a disassembly (removal) task in the AMM. Therefore as presented in previous section, separately for each of these parts a graph of disassembly can be generated.

In our separate graphs, it can happen that some of the tasks be common in each of these graphs. This is because some of the operation is same and shared between several tasks. This is why technical writers of AMM have assigned Task numbers to all of different operations in the AMM.

These common operations can be either in section of job setup (JSU) or process (P). For example in the job setup depressurizing a hydraulic cylinder can be common for many different task assemblies. Alternatively, in the section of process removing a panel can be a common operation for two different tasks if they are taking place around the same zone of airplane. When we are in the phase of EOL of an aircraft, when we have removed an access panel when performing task A, if also in procedures of task B, removal of access panel is required, we call this task a common task. Actually in AMM, when both of task A and task be call a same task number in their subtask, we can figure out that task is common and shared. With removing the access panel, which was in the procedure for performing task A, at the time of executing disassembly tasks for task B, this removal is already, done. Therefore, from a global point of view, for disassembly of multiple parts we will perform all the tasks in all the graphs so we can connect the common tasks in each of graphs. With doing this we will have a graph of disassembly for multiple parts.

Figure 3-11 demonstrates an example of disassembly graph for removal of two parts. Removal task A for disassembly of part A and removal task B for disassembly of part B. With using the task numbers and comparing them, we can find if in sequences of these two graphs there is a call to a same task. For instance let us assume that we do these comparison and it is seen that T12 has the same task number as T22. These two node will be the nodes that graphs can make a link. Therefore, with establishing this link we connect these two separate graphs together (Figure 3-12).

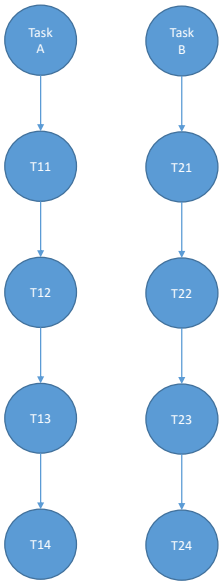


Figure 3-11: Disassembly graph for two tasks for removal of two parts

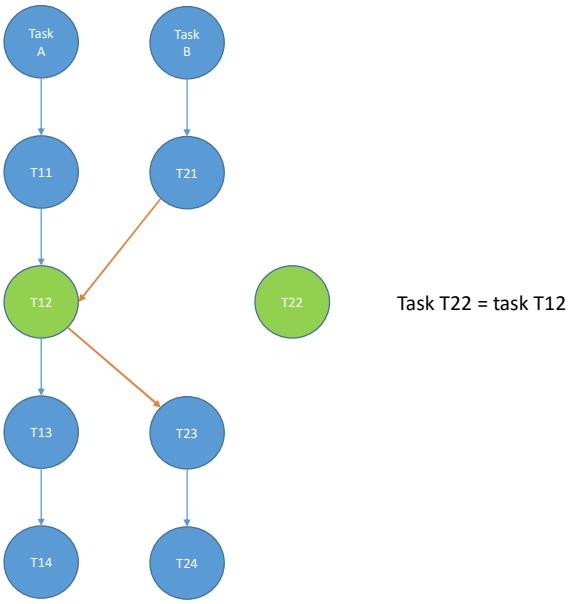


Figure 3-12: Connecting two graphs at the common tasks

Alternatively, in some cases it is also possible that the common operations for removal of two parts are not just common in just one task and the consecutive tasks are common tasks too. For instance

if the task T13 and task T23 be the same tasks, Figure 3-13 shows how two graphs are linked together.

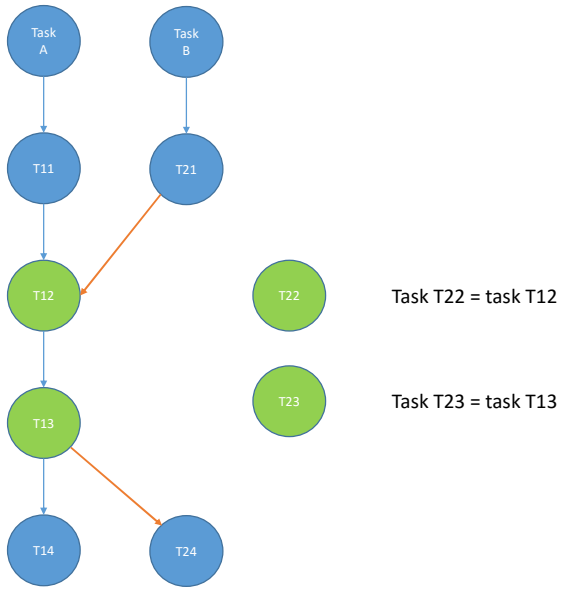


Figure 3-13: Connecting two graphs with consecutive common tasks

Similarly, if there are more parts this concept can be applied (Figure 3-14).

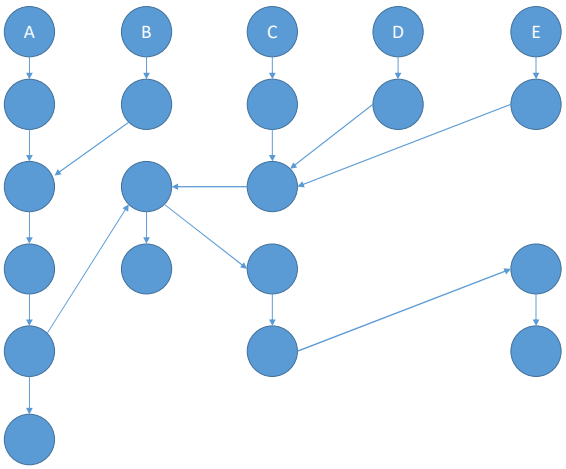
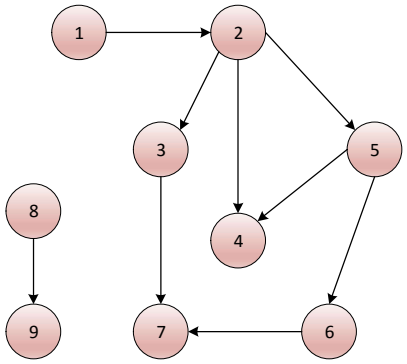


Figure 3-14: Connecting multiple graphs at the common tasks

3.3.3 Mathematical model of a disassembly graph

Adjacency matrix is a common method to mathematically represent graphs. For our case the principle is that with forming the disassembly graph for multiple parts, we will have a total number of "n" unique tasks in the graph, so we can make a matrix of (n x n) and assign a number starting from 1 and ending to n for each of tasks.

If task i is connected to task j, in adjacency matrix, related element of a_{ij} of matrix would get a 1 otherwise it would be a zero. Figure 3-15 illustrates an example of forming an adjacency matrix for a directed acyclic graph.



	1	2	3	4	5	6	7	8	9
1	0	1	0	0	0	0	0	0	0
2	0	0	1	1	1	0	0	0	0
3	0	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0
6	0	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1
9	0	0	0	0	0	0	0	0	0

Figure 3-15: Forming adjacency matrix of a Graph

3.4 Automatic generation of disassembly sequences

3.4.1 Data acquisition from AMM

As discussed earlier AMM consists of more than 45000 tasks. At the time of disassembly for an EOL aircraft, many tasks should be executed and tasks are dispersed in 3000 pages of AMM. Since task numbers are numerated in AMM, we can use this attribute to read from manual and find tasks, subtasks, and those tasks that are called in subtasks. (Mascle et al. 2014) describes how a Matlab code reads from PDF files of AMM CRJ 100/200:

"The function "Main" enters into the file, calculates the number of files, records the names of files in a variable and erases the useless files. The list of names is then transferred to the function "Analyse_files". This function sends the names of AMM files one by one to "Files_details" so that the latter sends back(dismisses) the various main tasks of the same AMM file as well as the location of the "JSUI", "JSU", "Procedure" and "Close-Out". With all this information, "Analyse_files" analyzes the subtasks and the tasks (sub-function "Subtask" which are in the paragraphs of "JSUI", "JSU", "Procedure" and "Close-Out"."

The output is saved in a variable named "data". Figure 3-16 shows structure of the data.

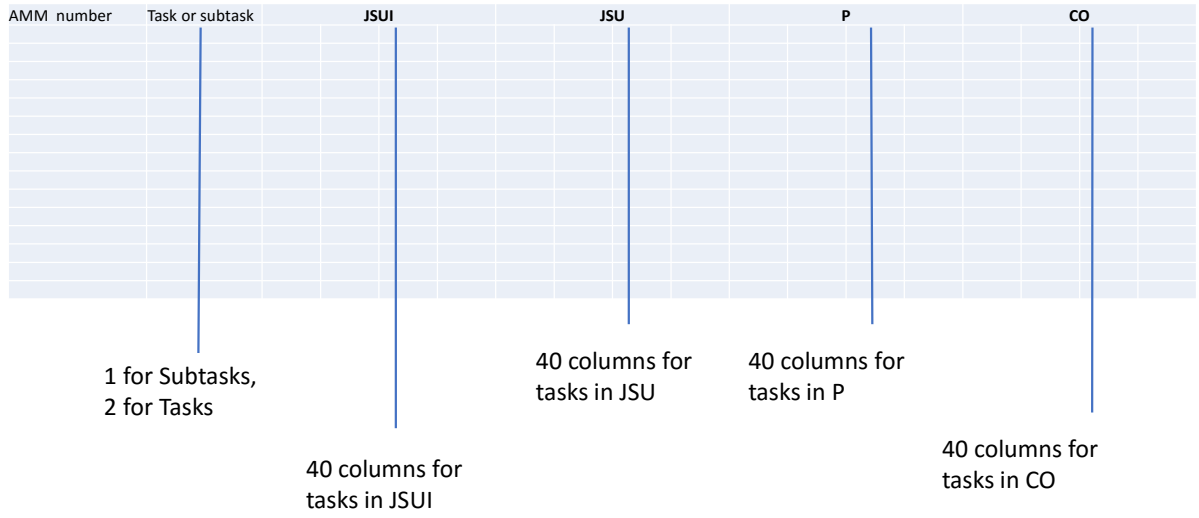


Figure 3-16: Structure of data generated from AMM

In the column of "AMM number", all of the tasks and subtasks in the AMM is listed. With looking up an AMM number, all the tasks associated with that AMM number can be found in its related row, which is divided and categorized in accordance with structure of AMM: JSUI, JSU, P and CO. For each of these sections, 40 columns has been considered, but not necessarily, all of the cells are filled. It is just the maximum space that is available. Depending on the task, these cells can be all empty or a few tasks or in some cases, more than ten tasks can be found in each section.

3.4.2 Automatic generation of disassembly sequences

For a given part, there should be a task number for removal/disassembly of that part in the AMM. Having the task number, a code can refer to databases of tasks described in Figure 3-16. With

performing a search and comparing the task number to tasks numbers in cells of “AMM number”, the related task and its related row can be found. Then all the subtasks and tasks which are for that task in AMM, is found in the associated row and can be listed. The found tasks are stored in a “cell” array. The property of a cell in Matlab permits storages of multiple of cells in a cell. A function named “subtask” serves this purpose.

As discussed earlier, JSUI section and close-out section is not required for our case which is disassembly of parts at EOL stage, therefore the “subtask” function does not look up in these two sections and won’t retrieve the related in those mentioned section.

Now we have a list of tasks and subtasks that is found and should be executed, however as described in section 3.3.1, the model needs to verify if these tasks and subtasks call other task. Therefore, if at first step for example 5 tasks/subtasks has been found, these 5 tasks/subtasks will be the input to “subtask” function, to find their associated tasks/subtasks (Figure 3-17). This iterative procedure continues to the point that all of tasks and subtasks are found and listed.

Management and recording the tasks according to the order found in the AMM is very important here. Not to lose the trace of sequences, right after subtasks are found, they are inserted into a graph. For management of this graph, a “tree class” in Matlab is utilized. Though the tree can have branches, we do not use that attribute and will just make a chain of nodes, which each node is a task. The root of the tree is a node that has no parents, so node is our “main task”. For other nodes, each node can only have one parent, which for our case is the precedence task and they do not have any siblings. The node can store any kind of data and a “string array” with name of task number with the format of xx-xx-xx-xxx-xxx is assigned to nodes. Nodes are mainly accessed through their index.

For the example in Figure 3-17, at first iteration, a chain is established. The parent is the main task (Node 0) and T1, T2, T3, T4 and T5 task numbers are assigned to node 1, 2, 3, 4, 5 accordingly.

At next iteration, first subtasks of T1 are found. These subtasks are stored in a branch with the class of tree. Then the tree is chopped at node 1, the chopped part are stored in a temporary branch. After that, the branch of subtasks of T1 is grafted to the tree and following it, the temporary branch is grafted at the end of the tree. With applying this procedure, we have followed principles of our model described in section 3.3.1 and sequences of execution of tasks are retained as specified in AMM. Figure 3-18 demonstrates this procedure.

With the same procedure, subtasks of tasks T2, T3, T4, and T5 should be retrieved and are grafted to the tree (which is actually a chain).

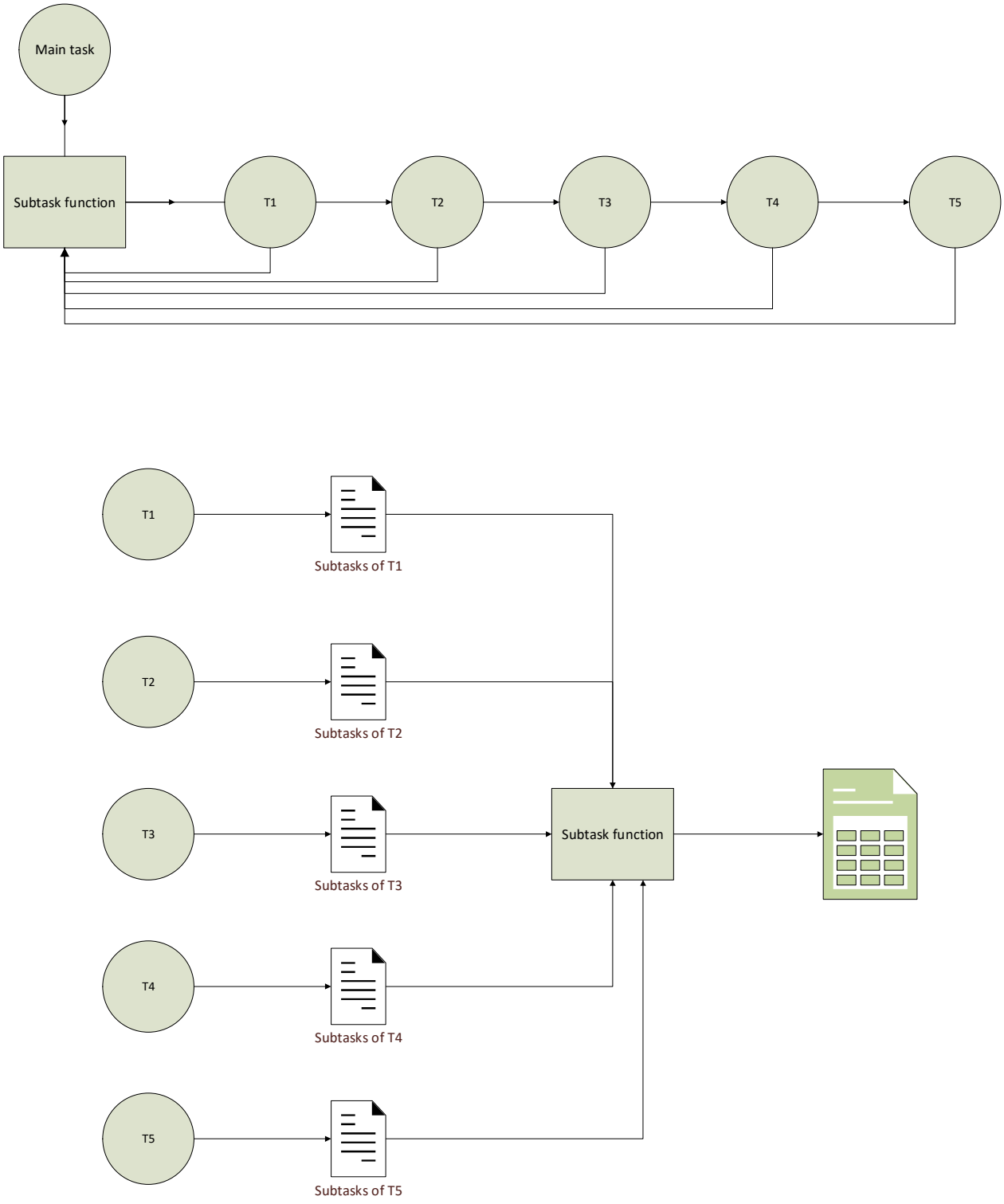


Figure 3-17: Iterative procedure of retrieving subtasks

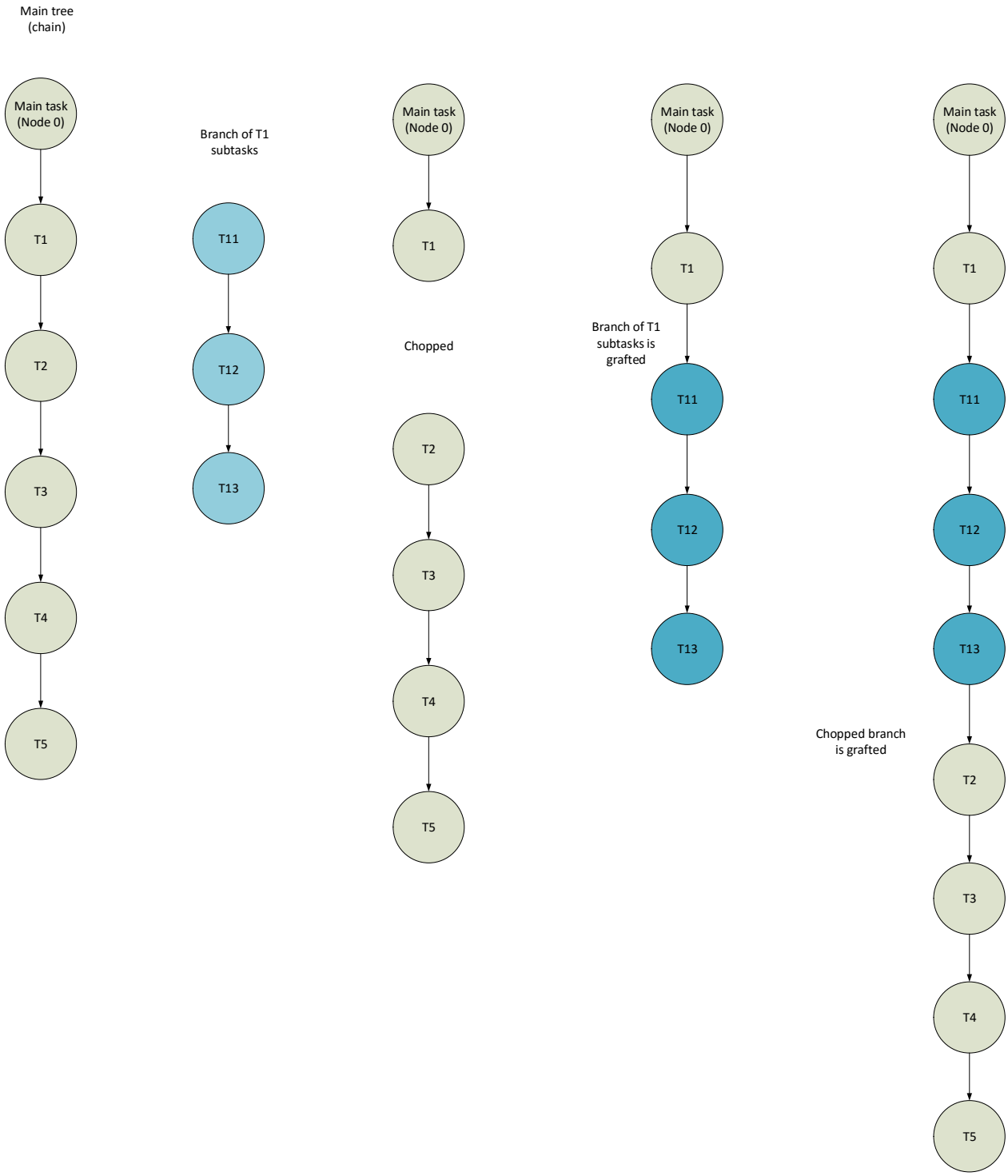


Figure 3-18: Procedure of establishing disassembly sequences for a Task

In an EOL aircraft, several parts are going to be disassembled. Then we will have several main tasks. Each of these tasks can be the input to the code and with the same procedure as described in this section, disassembly sequences of those tasks are generated and stored in a “tree class” variable

in Matlab. With having several parts as input to our code, consequently, we will have several trees in output. Each of these trees provides a sequence of disassembly for these parts.

3.4.3 Validation of model

To see how the developed model and code works, the best way is verifying it by referring to the Database that has read tasks from AMM as discussed in section 3.4.1. However to get sure that the database is also correct and for adding a second verification, at early stages of verification, to verify database, the AMM was a direct reference itself. By comparing the results (as output of code) with the AMM, an extra layer of Database verification is added to the validation of model. If the code read correctly from database but results was not in accord with AMM, the only possible reason would be that database has some bugs and is not in accord with AMM. If this was the case, feedbacks to developer of the data acquisition from AMM was sent to correct his code.

For the purpose of model validation, one can give any task as an input to the code and get subtasks from it, and then will refer to the Database/AMM to verify if the output is correct. When this verification is done for several any arbitrary tasks, it is concluded that the code will generate a correct output for any task.

In following example TASK 27−34−01−000−801 that is the task Removal of the Elevator is given to code and it automatically find all the subtasks and generates disassembly sequences automatically. In all of the iterations with referring to the AMM, correct output is verified.

1st iteration

Getting subs for: 27-34-01-000-801

The output is: 27-34-01-865-001, 27-34-01-941-002, 27-34-01-010-001,

27-34-01-490-002, 27-34-01-030-001, 27-34-01-020-001

2nd iteration:

Getting subs for: 27-34-01-865-001

Getting subs for: 27-34-01-941-002

Getting subs for: 27-34-01-010-001

Getting subs for: 27-34-01-490-002

Getting subs for: 27-34-01-030-001

Getting subs for: 27-34-01-020-001

Among these tasks, only 27-34-01-010-001 has subtasks and the output is:

55-12-01-000-801, 55-12-01-000-802, 55-23-01-000-801, 55-23-01-000-802

55-23-01-000-803, 55-23-01-000-804, 55-32-07-000-814

3rd iteration:

Getting subs for: 55-12-01-000-801

Output is: 55-12-01-941-001, 55-12-01-865-001, 55-12-01-020-001

Getting subs for: 55-12-01-000-802

Output is: 55-12-01-941-003, 55-12-01-865-003, 55-12-01-020-002

Getting subs for: 55-23-01-000-801

Output is: 55-23-01-863-001, 55-23-01-941-004, 55-23-01-865-001, 55-23-01-020-002

Getting subs for: 55-23-01-000-802

Output is: 55-23-01-863-002, 55-23-01-941-005, 55-23-01-865-003, 55-23-01-020-003

Getting subs for: 55-23-01-000-803

Output is: 55-23-01-863-003, 55-23-01-941-006, 55-23-01-865-005, 55-23-01-020-004

Getting subs for: 55-23-01-000-804

Output is: 55-23-01-863-004, 55-23-01-941-010, 55-23-01-865-007, 55-23-01-020-005

Getting subs for: 55-32-07-000-814

Output is: 55-32-07-863-014, 55-32-07-941-040, 55-32-07-865-027, 55-32-07-020-014

The iterations goes on in the same manner with finding subtasks for each task.

4th iteration:

Getting subs for: 55-12-01-941-001

Getting subs for: 55-12-01-865-001

Getting subs for: 55-12-01-020-001

Getting subs for: 55-12-01-941-003

Getting subs for: 55-12-01-865-003

Getting subs for: 55-12-01-020-002

Getting subs for: 55-23-01-863-001

Getting subs for: 55-23-01-941-004

Getting subs for: 55-23-01-865-001

Getting subs for: 55-23-01-020-002

Getting subs for: 55-23-01-863-002

Getting subs for: 55-23-01-941-005

Getting subs for: 55-23-01-865-003

Getting subs for: 55-23-01-020-003

Getting subs for: 55-23-01-863-003

Getting subs for: 55-23-01-941-006

Getting subs for: 55-23-01-865-005

Getting subs for: 55-23-01-020-004

Getting subs for: 55-23-01-863-004

Getting subs for: 55-23-01-941-010

Getting subs for: 55-23-01-865-007

Getting subs for: 55-23-01-020-005

Getting subs for: 55-32-07-863-014

Getting subs for: 55-32-07-941-040

Getting subs for: 55-32-07-865-027

Getting subs for: 55-32-07-020-014

5th iteration:

Getting subs for: 12-00-06-863-802

Getting subs for: 12-00-06-863-804

6th iteration:

Getting subs for: 12-00-06-869-001

Getting subs for: 12-00-06-910-002

Getting subs for: 12-00-06-863-003

Getting subs for: 12-00-06-863-004

Getting subs for: 12-00-06-869-002

Getting subs for: 12-00-06-910-004

Getting subs for: 12-00-06-863-006

7th iteration:

Getting subs for: 29-00-00-910-801

At this point, there are no other tasks called and iterations are stopped. Eventually the code generates disassembly sequences task 27-34-01-000-801 as following:

1. 27-34-01-865-001
2. 27-34-01-941-002
3. 27-34-01-010-001
4. 55-12-01-000-801
5. 55-12-01-941-001
6. 55-12-01-865-001
7. 55-12-01-020-001
8. 55-12-01-000-802
9. 55-12-01-941-003
10. 55-12-01-865-003
11. 55-12-01-020-002
12. 55-23-01-000-801
13. 55-23-01-863-001
14. 12-00-06-863-802
15. 12-00-06-869-001
16. 12-00-06-910-002
17. 29-00-00-910-801
18. 12-00-06-863-003
19. 12-00-06-863-004
20. 12-00-06-863-804
21. 12-00-06-869-002
22. 12-00-06-910-004
23. 12-00-06-863-006
24. 55-23-01-941-004
25. 55-23-01-865-001
26. 55-23-01-020-002
27. 55-23-01-000-802
28. 55-23-01-863-002
29. 55-23-01-941-005
30. 55-23-01-865-003
31. 55-23-01-020-003

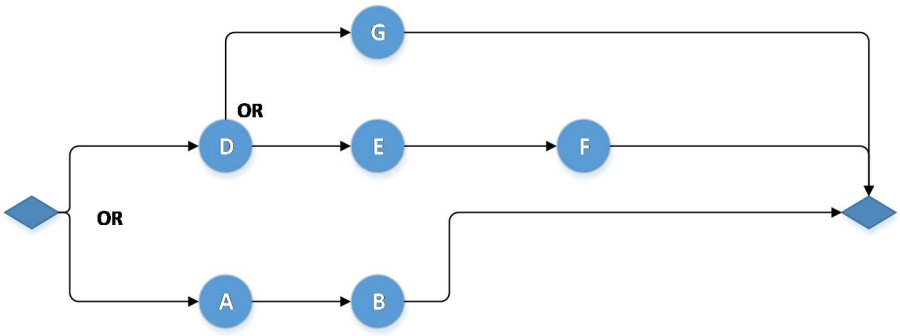
32. 55-23-01-000-803
33. 55-23-01-863-003
34. 55-23-01-941-006
35. 55-23-01-865-005
36. 55-23-01-020-004
37. 55-23-01-000-804
38. 55-23-01-863-004
39. 55-23-01-941-010
40. 55-23-01-865-007
41. 55-23-01-020-005
42. 55-32-07-000-814
43. 55-32-07-863-014
44. 55-32-07-941-040
45. 55-32-07-865-027
46. 55-32-07-020-014
47. 27-34-01-490-002
48. 27-34-01-030-001
49. 27-34-01-020-001

CHAPTER 4 DISASSEMBLY OPTIMIZATION

4.1 Introduction

The owner of a product at its EOL should decide which parts are in good state and have value to be used again and weather there are parts for being sent recycling facilities. The latter one can be performed using destructive approaches. However, when the parts are going to be used again, the process is commonly not destructive and is expressed as sequences of disassembly procedure. Usually there are many different ways for disassembly of a product. Disassembly optimization is involved with selecting an optimum sequence through set of all the possible disassembly sequences. The literature of disassembly sequencing implies optimization process is done via heuristics, metaheuristics, or mathematical programming.

Heuristic and metaheuristic methods usually return a 'good enough' solution. For having an exact optimum solution, mathematical programming is utilized but it deals with so much of computation and normally is for products with very few components since for determining all possible disassembly sequences, the search space tends to increase exponentially with the number of components (Lambert and Gupta, 2005). Figure 4-1 illustrates an example of a directed precedence graph with six parts.



ABDEFG	ADEFGB	DAEBFG	DEABFG	DEGABF
ABDEGF	ADEGBF	DAEBGF	DEABGF	DEGAFB
ABDGEF	ADEGFB	DAEFBG	DEAFBG	DEGFAB
ADBEFG	ADGBEF	DAEFGB	DEAFGB	DGABEF
ADEBGF	ADGEBF	DAEGBF	DEAGBF	DGAEBF
ADBGEF	ADGEFB	DAEGFB	DEAGFB	DGAEFB
ADEBFG	DABEFG	DAGBEF	DEFABG	DGEABF
ADEBGF	DABEGF	DAGEBF	DEFAGB	DGEAFB
ADEFBG	DABGEF	DAGEFB	DEFGAB	DGEFAB

Figure 4-1: Possible sequences for a directed precedence graph

It is seen that for a simple graph with just six components, there is 45 possible sequences. Thus for a case of disassembly with more parts there can be millions of possible sequences that is why heuristic or metaheuristic methods are widely used in finding an optimal (near optimal) disassembly procedure among many possible solutions.

For disassembly sequence optimization, researchers have chosen different parameters such as:

- Disassembly time for each part.
- Disassembly costs for each part.
- Tool change during disassembly.
- Disassembly direction changes.
- Revenue generated by part.
- Fastener types.
- Market demand of part.
- Combined multi-criteria.

Nevertheless, all the parameters eventually will lead to a disassembly project with less cost. For our case that is an aircraft at its end-of-life, none of the above parameters is available. We have the AMM and we have the tasks in AMM. In fact, a major difference with other works in literature shows up here. The other works focus on the disassembly sequencing of parts itself, but we have a task scheduling process that is constrained by its precedence. In each of our tasks, many parts might be mentioned to be disassembled and we do not have any information about quantity of these parts, duration of operations nor costs of disassembly or other parameters. Fortunately, as discussed in previous sections, ATA has standardized names of tasks with giving each task a task number. Task number helps us to specify zone of operation in the aircraft.

Since tasks are dispersed in different areas of aircraft, at the time of disassembly, different zones are visited. Thus, our optimization parameter is defined as number of total zone changes during a disassembly procedure.

For defining the zone of task numbers, the user can decide for depth of the zone using different elements of first three double digit of a task number: XX-XX-XX. If all these three elements are chosen, the zone depth will be at the device level and if first two element (XX-XX) is chosen, the zone depth will be at subsystem level.

With performing different simulations, it was observed that when the zone is chosen at device level, there is fewer chance that other tasks take place in that zone, however if the depth be chosen at

subsystem level, there is a good chance that a few other task take place in that zone. This will be in align with our target to have some other tasks in same zone so we can execute also those tasks. Eventually the author choses first two elements of a task number for assigning zones of operation to task number.

Figure 4-2 illustrates an example of a colored disassembly graph for disassembly of five parts generating five chains of disassembly sequences.

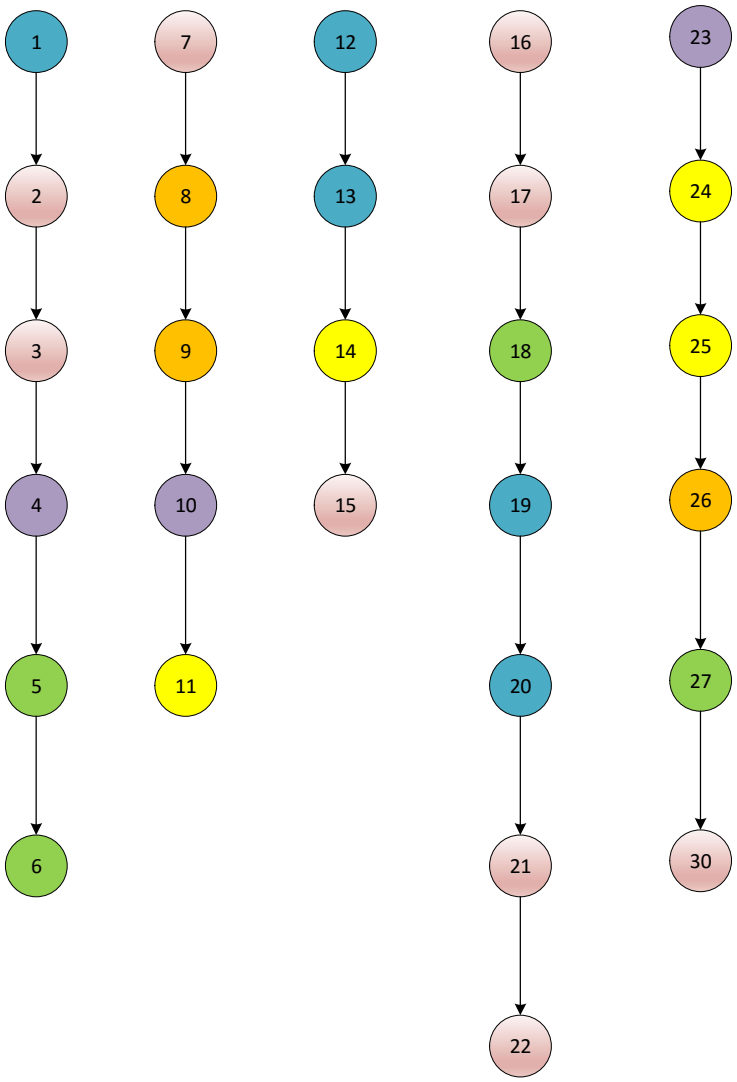


Figure 4-2: Assigning zones to disassembly tasks in disassembly model

In this example, in first instance of disassembly tasks 1 and 12, 7 and 16 are at the same zone though they are in different chains, thus our general concept for optimization is to perform task 12 also if we were directed to execute task 1 or similarly performing task 16 if disassembly was directed to execute task 7.

4.2 Graph theory

Graph theory is a very complex theory developed by mathematicians and has wide range of applications in engineering. One of the applications of graph theory is for finding shortest path problems. In this context, one of the most important and useful algorithm is Dijkstra's shortest path algorithm, that is an algorithm that finds shortest paths in a weighted acyclic directed graph.

The weighted graph is a graph that a weight is assigned on edges of the graph and actually, sum of weights of a path is the parameter that should be optimized. For instance, in google map for driving from a location to another location there can be several paths and on each edge between nodes of direction in the path, the distance between nodes can be assigned as the weight. In this example, Dijkstra's algorithm searches all possible routes and suggests an optimal path based on optimum total distance between departure and arrival nodes.

Acyclic graph is a graph that does not have any loops/cycles between its nodes. If cycles were existed in a graph, the algorithm falls into loops and actually will not be able to find any path.

Inspired by this shortest path problems in graph theory, the author tried to verify whether the disassembly model could be represented as an acyclic directed graph.

Since we have different parts at the time of disassembly of EOL aircraft, for some of the parts it is probable that their disassembly tasks call same tasks. This gives an idea to connect separate chains of disassembly sequences at the common tasks. Then instead of separate chains of disassembly sequences, an integrated graph is generated. If such this graph be generated all the precedence tasks of a common task would be executed and at the time of reaching the common task with performing the common task all of successors task will be available to be executed.

In this context, a Matlab code was developed to automatically generate the adjacency matrix of a disassembly graph with getting removal task number of two or more parts as inputs.

With analyzing the generated integrated graph for removal of different parts, it was concluded that generated graphs will not be acyclic and loops will exist in the graph.

Figure 4-3 illustrates how a loop is generated. In fact, this comes from structure of the tasks in AMM. For performing maintenance and for removal of parts at the time of maintenance, the technical writer actually has just taken into consideration the best procedures that is deemed feasible for removal of that part. Therefore, the focus has been for just that part not all the parts of aircraft. Apparently, at time of writing other procedures for other parts it has been possible to schedule some of tasks in different sequences for different parts at the discretion of technical writer.

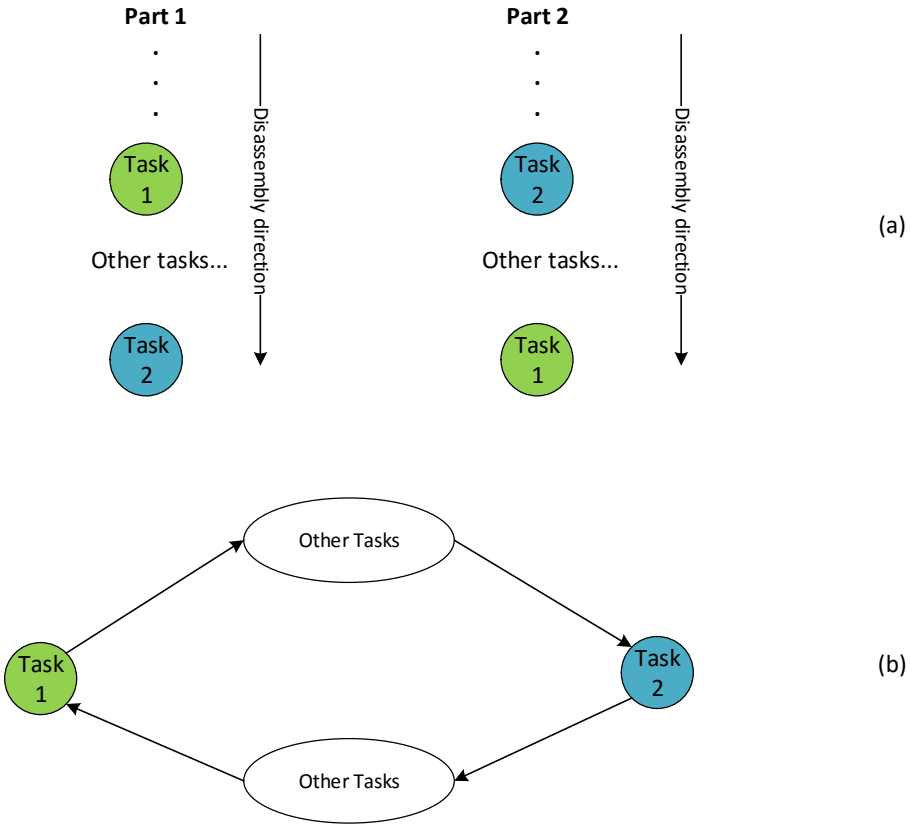


Figure 4-3: (a) Disassembly graph as separate chains (b) Forming of loops after integration of disassembly chains into one graph

Using real tasks of AMM and automatic generation of disassembly sequences and graphs could provide a bigger picture of the way tasks are scheduled in the AMM with more than 45000 tasks.

Eventually, the idea of using an integrated disassembly graph is overruled because of generating loops.

4.3 Heuristic methods

The developed Matlab code for this project automatically retrieves all tasks and subtasks that should be executed for disassembly of a part and generates a chain of disassembly sequences for each part. With using the code, we can arbitrary give one or several removal tasks as inputs to the code and check the quantity of tasks/subtasks that should be implemented. This will give an impression of the search space that we are dealing with. An example of a component that can be disassembled from aircraft is the engines. By referring to the AMM of CRJ100/200, it is observed that the removal procedure is a 62-pages procedure with so many subtasks that also call other tasks. In fact, the code generated a disassembly sequence chain with more than 700 tasks to be executed for just this single component. With adding disassembly sequence chain of some other parts, we have billions of feasible disassembly sequences that we need to choose one among them. This huge search space calls for a heuristic algorithm for determination of an optimal disassembly sequence.

4.3.1 Greedy algorithm

A greedy algorithm is an algorithm that selects best solution at an instance of a problem for the maximal immediate benefit, without considering how this selection affects future choices. A greedy algorithm works in phases. At each phase:

- It takes the best it can get right now, without regard for future consequences.
- It hopes that by choosing a local optimum at each step, it will end up at a global optimum.

Huffman encoding algorithm, Dijkstra's algorithm, Kruskal's algorithm and Prim's algorithm are examples of greedy algorithm.

This algorithm has also been widely used for disassembly sequencing problems. For example, (Han et al. 2013), utilized a "priority rule" or (Riggs et al., 2015) considered higher value parts for ruling the greedy algorithm. The "priority rule" has been used in other researches too. This method at each node for each of possible of paths, counts all the following nodes and selects a path with the more nodes in that path. For instance in Figure 4-2, node 16 has the highest priority with having more nodes after it to be executed; therefore, this node is selected to be executed first. In next

iteration, priorities for nodes are calculated again and next node is found for being executed is found.

For each case of disassembly, according to the configuration of graph and parameters that are considered for optimization, different rules can be applied in the greedy algorithm. For our case, the “priority rule” will not help that much because we have separate chains of disassembly and this rule just finds a chain with bigger size and executes its node till its size become less than second biggest chain and then implement one from 1st chain then the second chain till their size become less than third biggest chain.

Therefore, for our problem we define the greedy rule as following:

In each instance of disassembly, count tasks in each available zone and execute tasks in the zone that has highest number tasks.

Since our target is minimizing zone changes, when a task is implemented if the successor is still in the same zone, that successor can also be counted. This means that for example in Figure 4-2, for the case of Blue zone at the first instance, tasks 1 and 12 are available, but when we perform task 12, task 13 will also become available.

With considering this concept and counting successors tasks that are in the same zone and adding them to the tasks that we have right now available, we will have a total number of tasks which we can perform when we are in a zone. For example in Figure 4-2 we have three zones and if we go to a zone, below are the tasks that can be performed and that zone:

- 1st zone: tasks 1, 12, 13
- 2nd zone: tasks 7, 16, 17
- 3rd zone: task 23

Since 1st zone and 2nd zone both have same number of tasks either of zones can be considered for implementation. However, the algorithm goes for the first index, which is 1st zone, and adds tasks 1, 12 and 13 in disassembly path for implementation.

Order of execution of tasks of 1, 12, 13 in disassembly path can be any of following:

1→12→13

12→13→1

12→1→13

It is remarkable that in any of above orders of execution, the precedence constraint of disassembly graph is respected. (Which is for this case task 13 is done only if task 12 has been done before that). Therefore, we are looking for feasible disassembly sequences and we are free to choose among any of these orders, because our main concern is that a zone change does not happen.

At next iteration, tasks 2 and 14 become available for execution. Since task 3 is in the same zone as task 2, it is also counted in tasks that are executable in that zone without switching to another zone. It applies to task 17 too. Therefor at this instance we will have:

- 1st zone: tasks 2, 3, 7, 16, 17
- 2nd zone: task 14
- 3rd zone: task 23

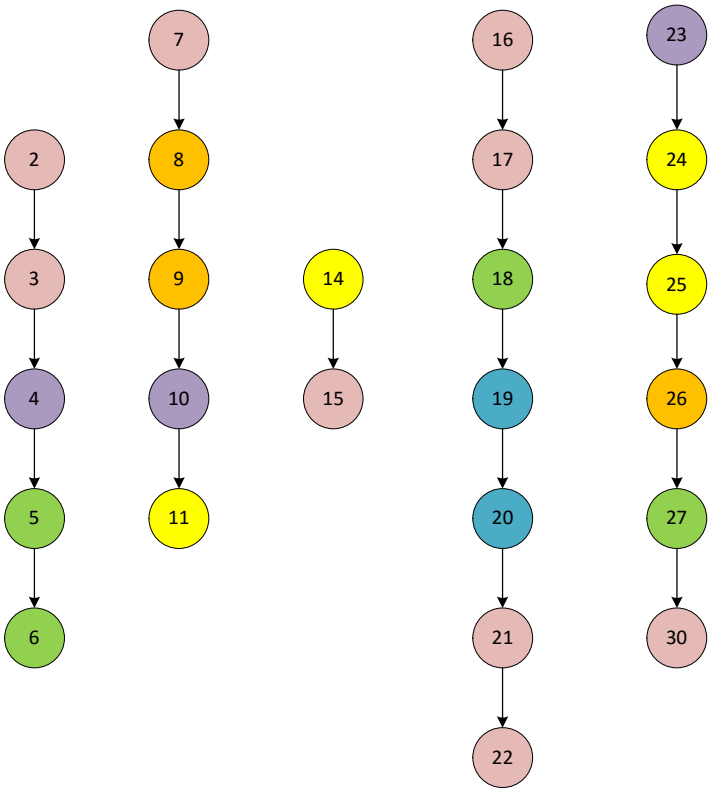


Figure 4-4: Update of Disassembly Graph at second iteration

Since 1st zone has higher number of tasks, therefore algorithm chooses this zone for implementation of its tasks at puts tasks 2, 3, 7, 16, 17 in disassembly path.

Next iteration (Figure 4-5):

1st zone: tasks 4, 23

2nd zone: tasks 8, 9

3rd zone: task 14

4th zone: task 18

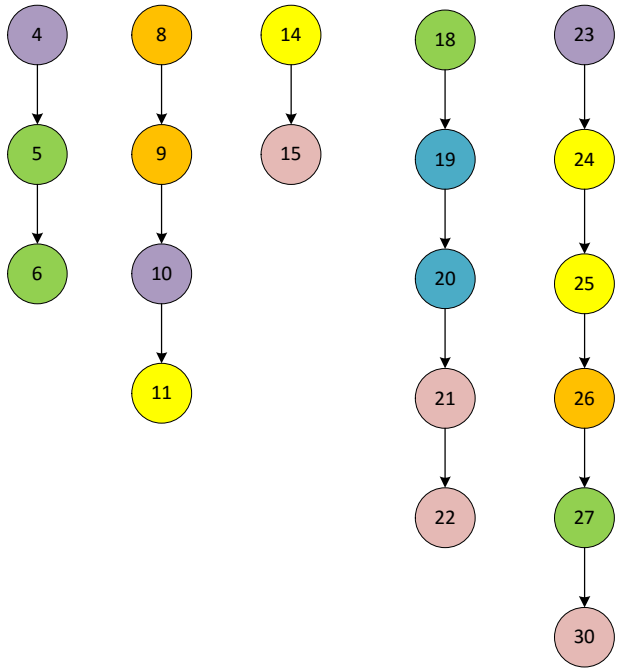


Figure 4-5: Update of Disassembly Graph at third iteration

At this instance, algorithm chooses tasks 4 and 23 for implementation and adds them in disassembly path.

The iterations goes on until all the nodes in the disassembly graph are visited and located in disassembly path. With same manner, algorithm for next iteration selects:

- Tasks 5, 6, 18
- Tasks 14, 24, 25

- Tasks 8, 9, 26
- Tasks 19, 20
- Tasks 15, 21, 22
- Task 10
- Task 11
- Task 27
- Task 30

Eventually the disassembly path generated by the greedy algorithm will be as following:

1 , 12, 13, 2, 3, 7, 16, 17, 4, 23, 5, 6, 18, 14, 24, 25, 8, 9, 26, 19, 20, 15, 21, 22, 10, 11, 27, 30

With this algorithm and using the code, a path with disassembly tasks is automatically generated.

4.3.2 Adaptive greedy algorithm

A modification to greedy algorithm is the adaptive greedy algorithm. This algorithm make decisions greedily/locally but tries to take into consideration some parameters from global point of view to help the decision to be more effective on global conditions.

In previous rule, we counted total number of executable tasks in each zone at each instance, no matter what are total number of tasks in that zone globally. Therefore switching to a zone with highest number of executable tasks in that zone does not have any attribute that how tasks in a zone are progressed.

For our adaptive greedy algorithm to choose between available zones, a parameter named AR is defined and utilized. The AR ratio is total number of executable tasks in a zone divided by total number of tasks to be done in that zone:

$$AR = \frac{Executable\ tasks\ in\ an\ availabe\ zone}{Total\ number\ of\ tasks\ in\ that\ zone} \tag{4.1}$$

For example in Figure 4-2 we will have:

- 1st zone: tasks 1, 12, 13 : (3 tasks can be done in this zone, without changing this zone)
  Total number of tasks in this zone: 5
- 2nd zone: tasks 7, 16, 17 : (3 tasks can be done in this zone, without changing this zone)
  Total number of tasks in this zone: 9
- 3rd zone: task 23 : (1 task can be done in this zone, without changing this zone)

  Total number of tasks in this zone: 3

Therefore, for calculating AR of each zone we will have:

- 1st zone: $AR_1 = 3/5 = 0.6$
- 2nd zone: $AR_2 = 3/9 = 0.333$
- 3rd zone: $AR_3 = 1/3 = 0.333$

At each iteration, the algorithm calculates AR for all of the available zones and executable tasks. Then tasks that are in the zone with maximum AR are selected to be implemented. Accordingly, the quantity of executed tasks in zone of operation is deducted from sum of total tasks in that zone. Therefore, value of "total number of tasks" in each zone is updated at each iteration.

For our example at this instance, AR_1 has the maximum value, therefore tasks 1, 12, 13 are selected to be executed and algorithm puts them on disassembly path.

For next iteration, we will have:

- 1st zone: tasks 2, 3, 7, 16, 17

  Total number of tasks in this zone: 9
- 2nd zone: task 14

  Total number of tasks in this zone: 4
- 3rd zone: task 23

  Total number of tasks in this zone: 3

Therefore, for calculating AR of each zone we will have:

- 1st zone: $AR_1 = 5/9 = 0.555$

- 2nd zone: AR_2= 1/4=0.25
- 3rd zone: AR_3=1/3=0.333

AR_1 has the maximum value, therefore tasks 2, 3, 7, 16, 17 are selected to be executed and algorithm puts them on disassembly path.

For next iteration, we will have:

1st zone: tasks 4, 23

Total number of tasks in this zone: 3

2nd zone: tasks 8, 9

Total number of tasks in this zone: 3

3rd zone: task 14

Total number of tasks in this zone: 4

4th zone: task 18

Total number of tasks in this zone: 4

Therefore, for calculating AR of each zone we will have:

- 1st zone: AR_1= 2/3=0.666
- 2nd zone: AR_2= 2/3=0.666
- 3rd zone: AR_3=1/4=0.25
- 4th zone: AR_4=1/4=0.25

AR_1 and AR_2 are equal and have the maximum values, so either tasks in 1st zone or 2nd zone can be chosen to be implemented. Our algorithm chooses the first zone and puts 4 and 23 in disassembly path.

Figure 4-6 shows the updated disassembly graph at this instance.

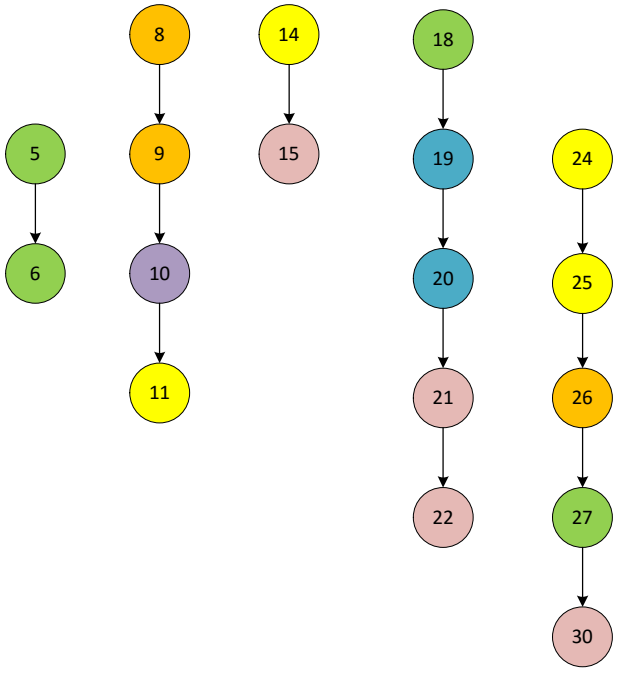


Figure 4-6: Update of Disassembly Graph at fourth iteration

For next iteration, we will have:

1st zone: tasks 5, 6, 18

Total number of tasks in this zone: 4

2nd zone: tasks 8, 9

Total number of tasks in this zone: 3

3rd zone: task 14, 24, 25

Total number of tasks in this zone: 4

Therefore, for calculating AR of each zone we will have:

- 1st zone: $AR_1 = 3/4 = 0.75$
- 2nd zone: $AR_2 = 2/3 = 0.666$
- 3rd zone: $AR_3 = 3/4 = 0.75$

AR_1 and AR_3 are equal and have the maximum values, so either tasks in 1st zone or 3rd zone can be chosen to be implemented. Our algorithm chooses the first zone and puts tasks 5, 6 and 18 in disassembly path.

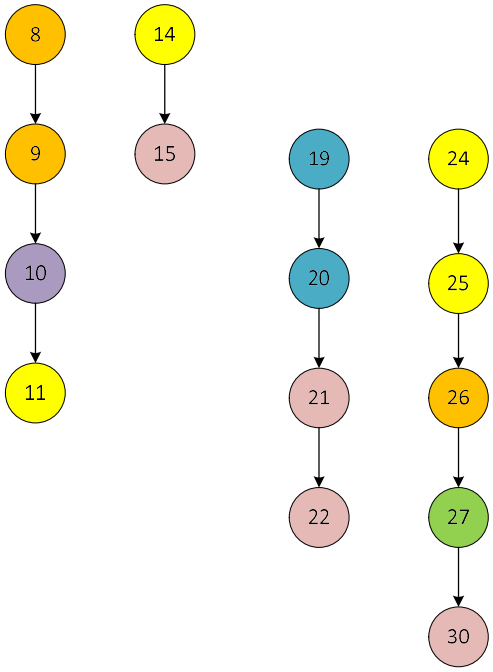


Figure 4-7: Update of Disassembly Graph at fifth iteration

For next iteration, we will have:

1st zone: tasks 8, 9

Total number of tasks in this zone: 3

2nd zone: tasks 14, 24, 25

Total number of tasks in this zone: 4

3rd zone: task 19, 20

Total number of tasks in this zone: 2

Therefore, for calculating AR of each zone we will have:

- 1st zone: $AR_1 = 2/3 = 0.666$
- 2nd zone: $AR_2 = 3/4 = 0.75$
- 3rd zone: $AR_3 = 2/2 = 1$

AR_3 has the maximum value, therefore tasks 19, 20 are selected to be executed and algorithm puts them on disassembly path.

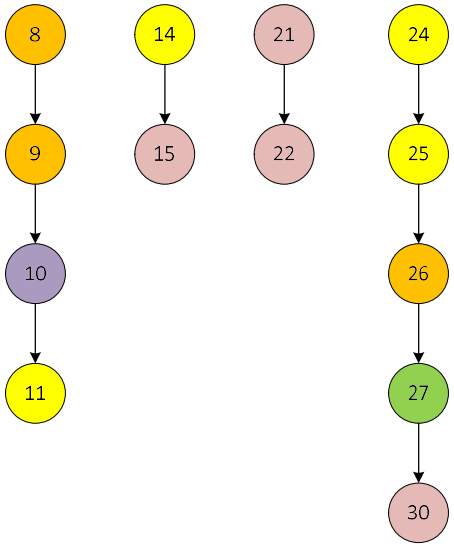


Figure 4-8: Update of Disassembly Graph at sixth iteration

For next iteration, we will have:

1st zone: tasks 8, 9

Total number of tasks in this zone: 3

2nd zone: tasks 14, 24, 25

Total number of tasks in this zone: 4

3rd zone: tasks 21, 22

Total number of tasks in this zone: 4

Therefore, for calculating AR of each zone we will have:

- 1st zone: $AR_1 = 2/3 = 0.666$
- 2nd zone: $AR_2 = 3/4 = 0.75$
- 3rd zone: $AR_3 = 2/4 = 0.5$

AR_2 has the maximum value, therefore tasks 14, 24, 25 are selected to be executed and algorithm puts them on disassembly path.

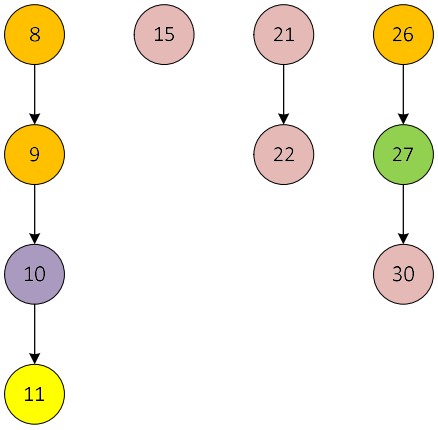


Figure 4-9: Update of Disassembly Graph at seventh iteration

For next iteration, we will have:

1st zone: tasks 8, 9, 26

Total number of tasks in this zone: 3

2nd zone: tasks 15, 21, 22

Total number of tasks in this zone: 4

Therefore, for calculating AR of each zone we will have:

- 1st zone: $AR_1 = 3/3 = 1$
- 2nd zone: $AR_2 = 3/4 = 0.75$

AR_1 has the maximum value, therefore tasks 8, 9, 26 are selected to be executed and algorithm puts them on disassembly path.

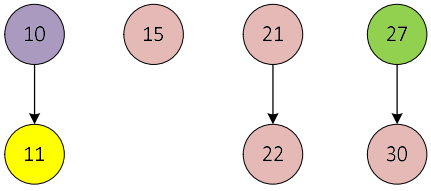


Figure 4-10: Update of Disassembly Graph at eighth iteration

With same manner, algorithm will choose tasks as following for next iterations:

- Task 10
- Task 11
- Task 27
- Tasks 15, 21, 22, 30

Eventually the disassembly path generated by the greedy adaptive algorithm will be as following:

1 , 12, 13, 2, 3, 7, 16, 17, 4, 23, 5, 6, 18, 19, 20, 14, 24, 25, 8, 9, 26, 10, 11, 27, 15, 21, 22, 30

In next section, comparison of these two discussed algorithms will be presented.

4.3.3 Comparison of results

For applying the proposed algorithms twenty different removal tasks are selected from AMM. The Matlab code automatically generates twenty separate chains of disassembly sequences. Now it is time for applying the optimization algorithms: 1- greedy algorithm 2- adaptive greedy algorithm.

Eventually, a disassembly sequence for all the given parts is automatically generated.

In other hand, a disassembly sequence without applying the optimization algorithm is also generated. For generating this disassembly sequences, each part is disassembled one by one from its start node to its end node and minimizing zone changes is not considered. This provides grounds for comparing the effects of optimization algorithms. Therefore, three feasible disassembly sequences is generated and each one is saved in variable named “route".

- Route 1: Disassembly sequences generated *without optimization.*
- Route 2: Disassembly sequences generated with *greedy algorithm.*
- Route 3: Disassembly sequences generated with *adaptive greedy algorithm.*

For the given 20 parts, the search space is around 12,000 tasks. The CPU time for this search space with applying greedy algorithms is approximately 14 min. After generation of routes, for each of routes, total number of zone changes is counted. Results is shown in

Table 4-1.

Table 4-1: Comparison of zone changes

Method	Total number of zone changes
Without Optimization	2567
Greedy Algorithm	1835
Adaptive Greedy Algorithm	976

Clearly, the applied *adaptive greedy algorithm* has been quite effective and works perfectly for our problem.

In our optimization problem from our graph of disassembly chains a graph with coloring of each node is established. The optimization was based on the number of colors, which actually resents number of tasks in a zone.

As discussed in previous sections, there is a possibility that in some of chains, common tasks exist in two or more chains. An approach is that when the algorithm determines a task to locate it in the route, that task be eliminated from all other chain if any. But since different operations maybe are done during disassembly operations and this is an automatic generation of disassembly task generation without human interference, a very conservative approach was applied. In this conservative approach, the algorithm does not remove the tasks located in disassembly route from other chains. This brings the opportunity to the operator to verify if that task with all the attributes is still in state of previous implementation. Therefore, instead of removing common tasks automatically in algorithm, the verification is left for discretion of disassembly technician. If all states and attributes of previous disassembly operation is still the same, actually, there is nothing left to be done and operator can simply go to the next sequence. However if this is not the case, the generated sequences is on the safe side. We can rest assured that any changes during disassembly operation with an effect on the previously implemented common task is foreseen in our procedure.

CHAPTER 5 CONCLUSION AND RECOMMENDATIONS

This chapter includes the contribution of this work. Recommendations for the future works are also listed.

5.1 Contributions

- ✓ Reviewing the result of the code for data acquisition from AMM. The provided database for this project had many several problems and with tedious manual verifications/checking of outputs and comparison with the data in AMM, the author provided feedback to the developer of code to remove bugs and correct his code.
- ✓ Studying the previous model developed by Camelot and outlining the problems.
- ✓ Developing a correct model for disassembly sequencing tasks in accordance with the structure and instructions of AMM. It was also pointed out that JSUI and CO sections of AMM is not required to be considered in retrieval of tasks.
- ✓ Developing a Matlab code for automatic generation of disassembly task sequences for any given removal task of AMM.
- ✓ Establishing an integrated digraph of tasks for disassembly of several parts of AMM with considering the proposed model and a Matlab code. The graph turned out to be a cyclic graph. For the problem of shortest path distances in graphs, the graph should be acyclic, consequently it is concluded utilizing graph theory for automatic determination of an optimized disassembly sequence is ruled out.
- ✓ Proposing and applying two algorithms for determination of an optimized disassembly sequencing with purpose of minimizing total number of changes in zones of operation in a disassembly project. With generating sequences automatically, it was observed that an adaptive greedy algorithm is quiet efficient for our problem.

5.2 Recommendations for future works

This work was done with a limited source of data in hand. The main constraint came from precedence relationships between tasks in the AMM. For reaching a better optimized disassembly

sequence plan, in the definition of fitness function that searches for optimized sequences, based on requirements of the project, other data such as following parameters from the case of an EOL aircraft should be integrated:

- Time of disassembly operations.
- Changes in direction of disassembly.
- Distance between zones of disassembly operations.
- Changes in tools.
- BOM of components.
- 3D models.
- Type of fasteners.
- Required skills of operators for different types of disassembly actions.
- Required equipment.
- Value of all of parts in the disassembly path for a selective disassembly.
- Priority ordered parts to supplied or reused.

It is hoped that this work encourages future studies in this subject to create a database for aforementioned parameters to be used for the application of disassembly sequence planning of EOL aircrafts.

Furthermore, most of algorithms in recent studies take advantage of evolutionary concept. Therefore, as first populations, some feasible solutions are provided for the algorithm. As much as the first populations are closer to a good optimal sequence, the inherited generations are better optimized. Consequently, results of this work provides inputs to those algorithms.

BIBLIOGRAPHY

Adenso-Diaz, B., Garcia-Carbajal, S., & Gupta, S. M. (2008). A path-relinking approach for a bi-criteria disassembly sequencing problem. *Computers & Operations Research, 35*(12), 3989-3997.

AFRA, Aircraft Fleet Recycling Association. Visited following link on 16/06/2016. http://afraassociation.org/

AFRA (2013). "Best Management Practice for Management of Used Aircraft and Assemblies and for Recycling of Aircraft Materials." Visited following link on 16/06/2016.

http://afraassociation.org/wp-content/uploads/2016/03/AFRA-BMP-3.2.pdf

Alshibli, M., El Sayed, A., Kongar, E., Sobh, T. M., & Gupta, S. M. (2016). Disassembly Sequencing Using Tabu Search. *Journal of Intelligent & Robotic Systems, 82*(1), 69-79.

Asmatulu, E., Twomey, J., & Overcash, M. (2013). Evaluation of recycling efforts of aircraft companies in Wichita. *Resources Conservation and Recycling, 80*, 36-45.

ATA iSpec 2200 Extract, ATA Standard Numbering System. Visited following link on 16/06/2016. https://www.ataebiz.org/specifications/

Baldwin, D.F., Abell, T.E., Lui, M.M., De Fazio, T.L., and Whitney, D.E., 1991, An integrated computer aid for generating and evaluating assembly sequences for mechanical products. *IEEE Transactions on Robotics and Automation*, 7, 78–94.

Behdad, S., Berg, L., Vance, J., & Thurston, D. (2014). Immersive Computing Technology to Investigate Tradeoffs under Uncertainty in Disassembly Sequence Planning. *Journal of Mechanical Design, 136*(7).

Berg, L. P., Behdad, S., Vance, J. M., & Thurston, D. (2015). Disassembly Sequence Evaluation: A User Study Leveraging Immersive Computing Technologies. *Journal of Computing and Information Science in Engineering, 15*(1).

Bourjault, A., 1984, *Contribution à une approche méthodologique de l'assemblage automatisé: elaboration automatique des séquences opératoires*, Ph.D. Thesis, Université de Franche-Comté.

Camelot, A. (2012). Modélisation des tâches pour la récupération de pièces réutilisables sur un avion en fin de vie. Master thesis, École polytechnique de Montréal.

Camelot, A., Baptiste, P., & Mascle, C. (2013). Decision Support Tool for the Disassembly of Reusable Parts on an End-of-Life Aircraft. *Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (IEEE-IESM 2013)*, 660-667.

Carrell, J., Zhang, H.-C., Tate, D., & Li, H. (2009). Review and future of active disassembly. *International Journal of Sustainable Engineering, 2*(4), 252-264.

Cheung, A., Ip, W.H. and Lu, D. (2005), Expert system for aircraft maintenance services industry. *Journal of Quality in Maintenance Engineering* 11(4), 348-58.

Cho, V., Wu, G., Ip, W. (2009). An Aircraft Service Staff Rostering using a Hybrid GRASP Algorithm. *International Journal of Engineering Business Management, 1(2)*, 13-18.

Christiand, & Yoon, J. (2007). Intelligent assembly/disassembly system with a haptic device for aircraft parts maintenance. *Computational Science - ICCS 2007, Pt 2, Proceedings, 4488*, 760-767.

Chung, C. and Peng, Q., (2006). Evolutionary sequence planning for selective disassembly in de-manufacturing. *International Journal of Computer Integrated Manufacturing* 19(3), 278-286.

CRIAQ-ENV412, (2011). Process for advanced management and technologies of aircraft end-of-life. Visited following link on 16/06/2016. http://www.polymtl.ca/env412/index.php

De Fazio, T.L. and Whitney, D.E., 1987, Simplified generation of all mechanical assembly sequences. *IEEE Journal of Robotics and Automation*, RA-3(6), 640–658.

Dong, J., Gibson, P., & Arndt, G. (2007). Disassembly sequence generation in recycling based on parts accessibility and end-of-life strategy. *Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture, 221*(6), 1079-1085.

Dong, T. Y., Zhang, L., Tong, R. F., & Dong, J. X. (2006). A hierarchical approach to disassembly sequence planning for mechanical product. *International Journal of Advanced Manufacturing Technology, 30*(5-6), 507-520.

Duta, L., & Douche, S. A. (2012). Dynamic Bayesian Network for Decision Aided Disassembly Planning. *Service Orientation in Holonic and Multi-Agent Manufacturing Control, 402*, 143-154.

Duta, L., Filip, F. G., & Popescu, C. (2008). Evolutionary programming in disassembly decision making. *International Journal of Computers Communications & Control, 3*, 282-286.

Duta, L., Filip, F. G., & Zamfirescu, C. (2008). Genetic Algorithms: a Decision Tool in Industrial Disassembly. *2008 Complexity & Intelligence of the Artificial & Natural Complex Systems, Medical Applications of the Complex Systems, Biomedical Computing*, 55-61.

ElSayed, A., Kongar, E., Gupta, S. M., & Sobh, T. (2012). An Online Genetic Algorithm for Automated Disassembly Sequence Generation. *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2011, Vol 3,* 657-664.

Erdos, G., Kis, T., and Xirouchakis, P., (2001). Modelling and evaluating product end-of-life options. *International Journal of Production Research* 39(6), 1203-1220.

Giri, R., & Kanthababu, M. (2015). Generating complete disassembly sequences by utilising two-dimensional views. *International Journal of Production Research, 53*(17), 5118-5138.

Giudice, F., & Fargione, G. (2007). Disassembly planning of mechanical systems for service and recovery: A genetic algorithm based approach. *Journal of Intelligent Manufacturing, 18*(3), 313-329.

Go, T. F., Wahab, D. A., Ab Rahman, M. N., Ramli, R., & Hussain, A. (2012). Genetically optimised disassembly sequence for automotive component reuse. *Expert Systems with Applications, 39*(5), 5409-5417.

Go, T. F., Wahab, D. A., Rahman, M. N. A., Ramli, R., & Azhari, C. H. (2011). Disassemblability of end-of-life vehicle: a critical review of evaluation methods. *Journal of Cleaner Production, 19*(13), 1536-1546.

Grochowski, D. E., & Tang, Y. (2009). A machine learning approach for optimal disassembly planning. *International Journal of Computer Integrated Manufacturing, 22*(4), 374-383.

Gungor, A., & Gupta, S. M. (1997). An evaluation methodology for disassembly processes. *Computers & Industrial Engineering, 33*(1-2), 329-332.

Gungor, A., & Gupta, S. M. (1998). Disassembly sequence planning for products with defective parts in product recovery. *Computers & Industrial Engineering, 35*(1-2), 161-164.

Gungor, A., & Gupta, S. M. (2001). Disassembly sequence plan generation using a branch-and-bound algorithm. *International Journal of Production Research, 39*(3), 481-509

Han, H. J., Yu, J. M., & Lee, D. H. (2013). Mathematical model and solution algorithms for selective disassembly sequencing with multiple target components and sequence-dependent setups. *International Journal of Production Research*, 51(16), 4997-5010.

Hassan, S., & Yoon, J. (2010). Haptic Guided Optimized Aircraft Maintenance Assembly Disassembly Path Planning Scheme. *International Conference on Control, Automation and Systems (ICCAS 2010)*, 1667-1672.

Hassan, S., & Yoon, J. (2014). Haptic assisted aircraft optimal assembly path planning scheme based on swarming and artificial potential field approach. *Advances in Engineering Software, 69*, 18-25.

Hsin-Hao, H., Wang, M.H., and Johnson, M.R., (2000). Disassembly sequence generation using a neural network approach. *Journal of Manufacturing Systems* 19(2), 73-82.

Iacob, R., Popescu, D., & Mitrouchev, P. (2012). Assembly/Disassembly Analysis and Modeling Techniques: A Review. *Strojniski Vestnik-Journal of Mechanical Engineering, 58*(11), 653-664.

Issaoui, L., Aifaoui, N., & Benamara, A. (2015). Solution space reduction of disassembly sequences generated automatically via computer aids. *Proceedings of the Institution of Mechanical Engineers Part C-Journal of Mechanical Engineering Science, 229*(16), 2977-2986.

Kanai, S. Takahashi, H. & Makino, H. (1996). ASPEN: Computer-aided Assembly Sequence Planning and Evaluation system based on predetermined time standard. *CIRP Annals - Manufacturing Technology* 45(1), 35-39.

Kang, J. G., & Xirouchakis, P. (2006). Disassembly sequencing for maintenance: a survey. *Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture, 220*(10), 1697-1716.

Kim, H. J., Lee, D. H., & Xirouchakis, P. (2007). Disassembly scheduling: literature review and future research directions. *International Journal of Production Research, 45*(18-19), 4465-4484.

Kim, H. J., Lee, D. H., Xirouchakis, P., & Kwon, O. K. (2009). A branch and bound algorithm for disassembly scheduling with assembly product structure. *Journal of the Operational Research Society, 60*(3), 419-430.

Kongar, E., & Gupta, S. M. (2006). Disassembly sequencing using genetic algorithm. *International Journal of Advanced Manufacturing Technology, 30*(5-6), 497-506.

Kuo, T.C., (2000). Disassembly sequence and cost analysis for electromechanical products. *Robotics and Computer-Integrated Manufacturing 16, 43–54.*

Lambert, A. J. D. (2002). Determining optimum disassembly sequences in electronic equipment. *Computers & Industrial Engineering* 43, 553-575.

Lambert, A. J. D. (2003). Disassembly Sequencing: A Survey. *International Journal of Production Research* 41(16), 3721–3759.

Lambert, A. J. D. (2006a). Exact methods in optimum disassembly sequence search for problems subject to sequence dependent costs. *Omega-International Journal of Management Science* 34(6): 538-549.

Lambert, A. J. D. (2006b). Generation of assembly graphs by systematic analysis of assembly structures. *European Journal of Operational Research* 168(3): 932-951.

Lambert, A. J. D. (2007). Optimizing disassembly processes subjected to sequence-dependent cost. *Computers & Operations Research* 34(2): 536-551.

Lambert, A.J.D. and Gupta, S.M., (2005). *Disassembly modeling for assembly, maintenance, reuse, and recycling*. Boca Raton, FL, CRC Press.

Lambert, A. J. D. and Gupta, S. M. (2008). *Methods for optimum and near optimum disassembly sequencing.* International Journal of Production Research 46(11): 2845-2865.

Langella, I. M. (2007). Heuristics for demand-driven disassembly planning. *Computers & Operations Research, 34*(2), 552-577.

Lazzerini, B., and Marcelloni, F. (2000). A genetic algorithm for generating optimal assembly plans. *Artificial Intelligence in Engineering, 14*(4), 319-329.

Li, J.R., Khoo, L.P., & Tor, S.B., (2005). An object-oriented intelligent disassembly sequence planner for maintenance. *Computers in Industry* 56(7), 699-718.

Liu, X. H., Peng, G. L., Liu, X. M., & Hou, Y. F. (2012). Disassembly sequence planning approach for product virtual maintenance based on improved max-min ant system. *International Journal of Advanced Manufacturing Technology, 59*(5-8), 829-839.

Mascle, C., and Balasoiu, B. A. (2001). Disassembly-assembly sequencing using feature-based life-cycle model. *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning (Isatp2001)*, 31-36.

Mascle, C., and Balasoiu, B. A. (2003). Algorithmic selection of a disassembly sequence of a component by a wave propagation method. *Robotics and Computer-Integrated Manufacturing, 19*(5), 439-448.

Mascle, C., Baptiste, P., Sainte Beuve, D., & Camelot, A. (2015). Process for Advanced Management and Technologies of Aircraft EOL. *12th Global Conference on Sustainable Manufacturing - Emerging Potentials, 26*, 299-304.

Mascle, C., CAI, Y. L. & Camelot, A., (2014). Information Technology for Processing and Treating Aircraft End of Life. *Applied Mechanics and Materials* 686, 153-159.

McGovern, S. M., & Gupta, S. M. (2006). Ant colony optimization for disassembly sequencing with multiple objectives. *International Journal of Advanced Manufacturing Technology, 30*(5-6), 481-496.

Mitrouchev, P., Wang, C. G., Lu, L. X., & Li, G. Q. (2015). Selective disassembly sequence generation based on lowest level disassembly graph method. *International Journal of Advanced Manufacturing Technology, 80*(1-4), 141-159.

Moore, K. E., Güngör, A. & Gupta, S.M. (1998). A Petri Net Approach to Disassembly Process Planning, *Computers & Industrial Engineering* 35(1–2), 165–168.

Moore, K. E., Güngör, A. & Gupta, S.M. (2001). Petri net approach to disassembly process planning for products with complex AND/OR precedence relationships. *European Journal of Operational Research* 135, 428-449.

Murata, T. (1989). Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE,* 77(4), 541 –580.

Murayama, T., Oba, F., Abe, S., & Yamamichi, Y. (2001). Disassembly sequence generation using information entropy and heuristics for component replacement. *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning (Isatp2001)*, 208-213.

O'Shea, B., Grewal, S. S., & Kaebernick, H. (1998). State of the art literature survey on disassembly planning. *Concurrent Engineering-Research and Applications, 6*(4), 345-357.

PAMELA. (2008). Airbus S.A.S.: *Process for Advanced Management of End-of- Life of Aircraft*. Company publication.

Popescu, D., & Iacob, R. (2013). Disassembly method based on connection interface and mobility operator concepts. *International Journal of Advanced Manufacturing Technology, 69*(5-8), 1511-1525.

Puente, S. T., Torres, F., Reinoso, O., & Paya, L. (2010). Disassembly planning strategies for automatic material removal. *International Journal of Advanced Manufacturing Technology, 46*(1-4), 339-350.

Rai, R., Rai, V., Tiwari, M. K., & Allada, V. (2002). Disassembly sequence generation: a Petri net based heuristic approach. *International Journal of Production Research, 40*(13), 3183-3198.

Reveliotis, S. A. (2007). Uncertainty management in optimal disassembly planning through learning-based strategies. *Iie Transactions, 39*(6), 645-658.

Ribeiro, J. S., & Gomes, J. D. (2015). Proposed framework for End-Of-Life aircraft recycling. *12th Global Conference on Sustainable Manufacturing - Emerging Potentials, 26*, 311-316.

Rickli, J. L., & Camelio, J. A. (2014). Partial disassembly sequencing considering acquired end-of-life product age distributions. *International Journal of Production Research, 52*(24), 7496-7512.

Riggs, R. J., & Hu, S. J. (2013). Disassembly liaison graphs inspired by word clouds. *Forty Sixth CIRP Conference on Manufacturing Systems 2013, 7*, 521-526.

Riggs, R. J., Jin, X. N., & Hub, S. J. (2015). Two-stage sequence generation for partial disassembly of products with sequence dependent task times. *22nd CIRP Conference on Life Cycle Engineering, 29*, 698-703.

Sarin, S. C., Sherali, H. D., & Bhootra, A. (2006). A precedence-constrained asymmetric traveling salesman model for disassembly optimization. *Iie Transactions, 38*(3), 223-237.

Seo, K.K., Park, J.H., & Jang, D.S., (2001). Optimal disassembly sequence using genetic algorithms considering economic and environmental aspects. *The International Journal of Advanced Manufacturing Technology* 18(5), 371-380.

Shimizu, Y., Tsuji, K., & Nomura, M. (2007). Optimal disassembly sequence generation using a genetic programming. *International Journal of Production Research, 45*(18-19), 4537-4554.

Smith, S., Smith, G., & Chen, W. H. (2012). Disassembly sequence structure graphs: An optimal approach for multiple-target selective disassembly sequence planning. *Advanced Engineering Informatics, 26*(2), 306-316.

Srinivasan, H. and Gadh, R. (1998). A geometric algorithm for single selective disassembly using the wave propagation abstraction. *Computer-Aided Design*, 30(8), 603-613.

Srinivasan, H. and Gadh, R. (2000). Efficient geometric disassembly of multiple components from an assembly using wave propagation. *Journal of Mechanical Design,* 122, 179-184.

Srinivasan, H. and Gadh, R. (2002). A non-interfering selective disassembly sequence for components with geometric constraints. *IIE Transactions,* 34, 349-361.

Srinivasan, H., Figueroa, R. and Gadh, R. (1999). Selective disassembly for virtual prototyping as applied to de-manufacturing. *Robotics and Computer-Integrated Manufacturing*, 15, 231-245.

Takeyama H., Sekiguchi, H., T. Kojima, T., Inoue, K., and Honda, T. (1983). Study on Automatic Determination of Assembly Sequence. *CIRP Annals-Manufacturing Technology* 32(1), 371–374.

Tang, Y., Zhou, M. C., Zussman, E., & Caudill, R. (2002). Disassembly modeling, planning, and application. *Journal of Manufacturing Systems, 21*(3), 200-217.

Tian, G. D., Zhou, M. C., & Chu, J. W. (2013). A Chance Constrained Programming Approach to Determine the Optimal Disassembly Sequence. *I Transactions on Automation Science and Engineering, 10*(4), 1004-1013.

Torres, F., Puente, S. T., & Aracil, R. (2003). Disassembly planning based on precedence relations among assemblies. *International Journal of Advanced Manufacturing Technology, 21*(5), 317-327.

Torres, F., Puente, S., & Diaz, C. (2009). Automatic cooperative disassembly robotic system: Task planner to distribute tasks among robots. *Control Engineering Practice, 17*(1), 112-121.

Tripathi, M., Agrawal, S., Pandey, M. K., Shankar, R., & Tiwari, M. K. (2009). Real world disassembly modeling and sequencing problem: Optimization by Algorithm of Self-Guided Ants (ASGA). *Robotics and Computer-Integrated Manufacturing, 25*(3), 483-496.

Wan, H. D., & Gonnuru, V. K. (2013). Disassembly planning and sequencing for end-of-life products with RFID enriched information. *Robotics and Computer-Integrated Manufacturing, 29*(3), 112-118.

Wang, C. G., Mitrouchev, P., Li, G. Q., & Lu, L. X. (2016). Disassembly operations' efficiency evaluation in a virtual environment. *International Journal of Computer Integrated Manufacturing, 29*(3), 309-322.

Wang, H., Xiang, D., & Duan, G. H. (2008). A genetic algorithm for product disassembly sequence planning. *Neurocomputing, 71*(13-15), 2720-2726.

Wang, H., Xiang, D., Rong, Y. M., & Zhang, L. X. (2013). Intelligent disassembly planning: a review on its fundamental methodology. *Assembly Automation, 33*(1), 78-85.

Xia, K., Gao, L., Li, W. D., & Chao, K. M. (2014). Disassembly sequence planning using a Simplified Teaching-Learning-Based Optimization algorithm. *Advanced Engineering Informatics, 28*(4), 518-527.

Yeh, W. C. (2012). Simplified swarm optimization in disassembly sequencing problems with learning effects. *Computers & Operations Research, 39*(9), 2168-2177.

Yun, Y., & Moon, C. (2011). Genetic algorithm approach for precedence-constrained sequencing problems. *Journal of Intelligent Manufacturing, 22*(3), 379-388.

Zhang, H. C., Kuo, T. C. (1997). *A graph-based disassembly sequence planning for EOL product recycling*. 21st IEEE/CPMT International Electronics Manufacturing Technology (IEMT) Symposium, 140-151.

Zhang, K. F., Zhao, L., Li, Y., & Shao, Y. (2008). Effective Component Disassembly Approach for Aircraft Assembly Based on Fuzzy-Clustering Algorithm. *2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 1-3*, 850-855.

Zhao, S. E., Li, Y. L., Fu, R., & Yuan, W. (2014). Fuzzy reasoning Petri nets and its application to disassembly sequence decision-making for the end-of-life product recycling and remanufacturing. *International Journal of Computer Integrated Manufacturing, 27*(5), 415-421.