Workload Balancing in Transportation Crew Scheduling

by

Fardin Dashty Saridarq

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University
Summer 2013

Qaradağ deprəmində yaşamını itirən kimsəsizlərə sunulur...

Acknowledgements

I owe my deepest gratitude to my advisor, Dr. Güvenç Şahin, for his enthusiasm, encouragement and guidance throughout my graduate studies. It has been and always will be a great pleasure to work with him. Besides Dr. Şahin, I would like to thank Dr. Kemal Kılıç, Dr. Abdullah Daşçı, Dr. Tonguç Ünlüyurt, Dr. Gündüz Ulusoy, Dr. John R. Current and Dr. Hans Frenk for influencing me to higher education, to research, and to teaching. I could not have asked for better role models, each inspirational, supportive and patient.

One of the joys of completion is to look over the past and remember all the friends and family who have helped and supported me along this short but fulfilling journey. I am indebted to my many student colleagues for providing a stimulating and fun environment in which it is pleasure to learn and grow. I would like to express my heartiest thanks to all Azerbaijani students of the Sabanci University and every single member of FENS-1021. Finally and above all, I would like to thank my family for their love, patience and support at all times.

This research has been supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) under Grant 110M495.

WORKLOAD BALANCING IN TRASPORATION CREW SCHEDULING

Fardin Dashty Saridarq

Industrial Engineering, Master's Thesis, 2013

Thesis Supervisor: Assoc. Prof. Dr. Güvenç Şahin

Keywords: Crew Scheduling, Operational Planning, Workload Balancing, Cost Expanded Network, Network Flow

Abstract

We focus on workload balancing in crew scheduling problems of transportation systems where deadheading (repositioning with no duty) of crew is also possible. The deadheading option could be just used unnecessarily for the sake of balancing the workload among team members. Existing works have only focused on systems where deadheading is not considered. The assignment of crew members to a sequence of duties in a finite planning horizon is determined in such a way that the allocation of the workload among the crew members is acceptably fair and almost equal when possible. This issue is a common planning phenomenon for also other type of duty scheduling and rostering processes where teams of crew members are in consideration such as hospitals and airlines. At the tactical level, the crew schedules are feasible with respect to various restrictions and regulations; yet, they may result in an imbalanced share of workloads among the crew. In addition, unbalanced crew schedules may also cause unavoidable over-time costs and result in unevenness with respect to time-based compensations. A solution approach based on a network flow formulation of the problem is developed. In addition, we develop a binary search method as an exact algorithm and a pool of conventional heuristic methods that modify the schedules by reallocating the duties without disrupting the feasibilities. We present the results of our computational experiments with well-known problem instances from the crew scheduling literature and data sets that are representative of largest crew region in Turkish State Railways.

ULAŞIM SİSTEMLERİNİN EKİP PLANLAMA PROBLEMLERİNDE İŞ YÜKÜ DENGELEME

Fardin Dashty Saridarq
Endüstri Mühendisliği, Yüksek Lisans Tezi, 2013
Tez Danışmanı: Doç. Dr. Güvenç Şahin

Anahtar Kelimeler: Ekip Planlama, Operasyonel Düzeyde Planlama, İş Yükü Dengeleme, Maliyet Bazlı Genişletilmiş Çizge, Çizge Akış

Özet

Bu çalışmada ulaşım sistemlerinin ekip planlama problemlerindeki iş yükü dengeleme konusu ele alınmıştır. Bu sistemlerde ekip üyelerini görevsiz konumlandırma (gerekli olmadığı bir yerden gerekli olduğu bir yere her hangi bir göreve atanmadan gönderilmesi) da olasıdır. Bu seçenek, ekip üyeleri arasında iş yükünü dengeli bir şekilde dağıtma amacıyla gereksiz yere kullanılabilir. Dengeli iş yükü dağıtımı problemini ele alan mevcut çalışmalar görevsiz konumlandırma olasığı olmayan sistemlere odaklanmıştır. Sonlu bir planlama ufkunda ekip üyelerinin görev dizilerine ataması yapılırken, personel arasında iş yükü tahsisi kabul edilebilir seviyede adil ve mümkün olduğunca hemen hemen eşit bir şekilde belirlenir. Bu konu, hastaneler ve havayolları gibi görev planlama ve görev atama süreçleriyle uğraşan işletmeler için de önemli bir planlama meselesidir. Taktik planlama düzeyinde, ekip çizelgeleri çeşitli kurallar ve kısıtlar açısından uygun bir şekilde yapılabilir; ancak, bu çizelgeler çalışanlar arasında iş yükünün dengesiz dağılımına neden olabilir. Buna ek olarak, dengesiz bir ekip programı kaçınılmaz fazla mesai maliyetlerine yol açabilir ve çalışm saatlerine bağlı tazminatların dağılımı açısından adaletsiz olabilir. Ele aldığımız iş yükü dengeleme problemi için kesin çözüm yöntemleri olarak bir ağ akış problemi gösterimi ve bir de ikili arama yöntemi geliştirildi. Sezgisel yöntemler olarak ise konvansiyonel operatörler kullanılarak, görev çizelgelerinin olurluluklarını koruyarak görevleri yeniden tahsis eden yerel komşuluk arama algoritmaları geliştirildi. Hesaplamalı deney sonuçları ekip planlama literatürünün iyi bilinen problem örnekleri ve Türk Devlet Demiryolları'nın en büyük ekip bölgelerinden alınan veri setleri için sunulmuştur.

TABLE OF CONTENTS

List of Figures

Chapter 1

Introduction

Crew-related costs have a significant share in transportation system costs. Especially in railways due to complex physical infrastructure and interrelated operations, this cost constitutes a high portion of the expenditure. Abbink et al. (2007) predicts that a new planning system would reduce costs of Dutch Railways by 6 million Euros. This huge amount of reduction is due to large amount of crew related costs of more than 3000 drivers working in 29 crew regions, covering more than 1000 duties using 14000 timetabled daily trains.

In most systems, there are two types of crew related costs. Firstly, a crew member is paid a fixed monthly salary. In addition, there is a time-based compensation that depends on the workload of the crew. Assignment of different types of duties to a crew member may also affect the time-based compensation. A railway system is usually composed of multiple and sometimes many crew regions that are mostly independent from each other from a managerial point of view. However, the system requires both coordination and cooperation among these regions in order to successfully execute the operations that are imposed by a central authority (i.e. the headquarters). Considering the importance of crew resources in operations their significant share of costs necessitates a traditional hierarchical decision making process which may be structured as follows:

- At a strategic planning level, system-wide issues are considered. Regulations enforced by the labor unions and politics of the company are evaluated at this level.

Determining the number of regions and their spread over the system, locations of the home stations, allocation of train duties among different regions and specifying the locations of crew exchange stations are some examples of strategic level problems.

- Tactical planning is usually concerned with regional problems related to managing crew members in each crew region. For example, determining the minimum required crew capacity of each region, i.e. the minimum sufficient crew resources level for one crew region required to operate a given list of train duties is determined at this level. Considering the significant portion of crew related costs in railway companies, this problem is very important to decrease the total crew-related costs.
- Operational planning is related to managing daily operations. Similar to the tactical planning, each crew region is considered independently. Crew scheduling problems at the operational level are concerned with final assignment of crew members to duties during a finite short planning horizon. Managerial issues such as fairness in duty assignments and balancing the workload (and associated payments) among the crew are important planning issues at this level.

We study the workload balancing problem in transportation crew scheduling where fairness issue is studied in terms of the work hours of crew members. Fairness has been considered in personnel scheduling literature, and in particular, in nurse rostering problems. In order to minimize the time-based compensation, it is required to maintain fairness with respect to working conditions for all crew members. This issue is a common planning phenomenon for also other type of duty scheduling and rostering processes where teams of crew members are in consideration such as railways and airlines.

At the tactical level, the planning problem is concerned with determining the sufficient minimum capacity of a crew region in order to operate the assigned train schedule. The size of the crew, i.e. number of crew members, in a region is determined at this level. Although the resulting crew schedules are feasible with respect to various restrictions and regulations, they may not satisfy the comfort and requests of the crew members particularly due to an imbalanced share of workload among them. In addition, unbalanced crew schedules may also cause unavoidable over-time costs and result in unevenness with respect to time-based compensations. The workload balancing problem is concerned with the assignment of crew members to a sequence of duties in a finite planning horizon in such a way that the allocation of the workload among the crew members is acceptably fair and almost equal when possible.

Our contributions in this study can be summarized as follows:

- We define the workload balancing problem in transportation crew scheduling; we formulate a mathematical programming problem considering all rules and restrictions.
- An exact algorithm is designed using a binary search method:
  - various improvements on the search mechanism of the binary search algorithm are presented;
  - properties of the optimal solution of the problem are explored.
- Multiple conventional heuristic methods based on a neighborhood search idea are developed.
- In order to compare the efficiency and effectiveness of designed algorithms, we perform a computational study with well-known problem instances from the crew scheduling literature and data sets that are representative of largest crew regions in Turkish State Railways.

Following a review of the literature on crew-related fairness and balancing problems in Chapter 2, we present our study on the workload balancing problem in transportation crew scheduling in detail in Chapter 3. We develop an exact algorithm using a binary search method in Chapter 4. In Chapter 5, we present a pool of conventional heuristic algorithms based on a neighborhood search idea. Our computational study is presented in Chapter 6. And finally in Chapter 7, we conclude with a summary and some remarks on future research.

Chapter 2

Literature Review

There is very limited number of studies concerned with the workload balancing problem in transportation crew scheduling literature. Burke et al. (2001) consider the fairness issue constraint as a soft constraint for a nurse rostering problem. This constraint ensures distributing the duty types -morning, night, waiting shifts, etc.- uniformly over the personnel with the same work regulation. Bellanti et al. (2004) introduce evenly assigned working shifts and days off during the weekends as well as the balanced assignment of morning, afternoon and night working shifts as operational requirements. Employee timetabling problems with flexible workload are studied by Chiarandini et al. (2000). They define positive and negative flexibility for each employee. A positive flexibility is number of weeks in which the employee worked more than a fixed workload while negative flexibility is number of weeks in which the employee worked less than a fixed workload. Two components of the objective function consists of balancing positive flexibility and negative flexibility by keeping positive and negative values as uniform as possible for all employees.

Cappanera and Scutellà (2005) consider the problem of finding k balanced paths from a source node to a destination node in a weighted acyclic network, where the difference in cost between the longest and the shortest path is minimized. Cappanera and Scutellà (2005) propose exact and approximate algorithms for node-disjoint and arc-disjoint versions of the problem. This problem on a weighted network is presented by Cappanera and Scutellà (2011). They focus on computing node-disjoint balanced paths in general case, where the associated network could have any structure. A pool of algorithms which includes an exact as well as alternative heuristics based on the color-coding method is designed.

In the context of crew planning problems, Şahin and Yüceoğlu (2011) study the tactical crew capacity planning in railways which involves finding the minimum number of crew members to cover all duties in a planning horizon called the minimum crew capacity problem. They develop a sequential and an integrated approach based on a time-space network representation. Moreover, Suyabatmaz and Şahin (2011) develop a set-covering type formulation for the minimum crew capacity problem and propose a column-and-row generation algorithm for the resulted problem formulation. Both of these studies focus on a tactical planning level. In our work, we suppose that the results of the tactical planning level could be an input as we focus on a more operational level where the decisions are mostly concerned with the final assignment of duties to crew.

From a methodological point of view, our work follows the footsteps of both Cappanera and Scutellà (2011) and Şahin and Yüceoğlu (2011). The balanced bath problem in Cappanera and Scutellà (2011) works with node-disjointness of the path. This idea, however, was not directly applicable on the more generic network representation in Şahin and Yüceoğlu (2011). In particular, the existence of deadheading in transportation crew schedules necessiates a further adaptation of the modelling approach in Cappanera and Scutellà (2011) according to the network representation in Şahin and Yüceoğlu (2011).

Chapter 3

Problem Definition

In this study, we consider the workload balancing problem (WBP) as to balance the workload of railway crew in a region by minimizing the difference between the maximum workload assigned to a crew member and the minimum workload assigned to a crew member during a finite planning horizon. This problem should be considered as an operational level planning problem for a railway region during the final rostering phase of the crew scheduling process while the crew members are assigned to their duties given the available number of crew members (i.e. predetermined crew capacity of the region). In order to formulate our problem we use the network representation and the network flow model for the crew capacity planning problem (CCPP) suggested by Şahin and Yüceoğlu (2011). From the methodological point of view, we benefit from the ideas and the solution method for the balanced path problem (BPP) introduced by Cappanera and Scutellà (2011).

3.1. The Balanced Path Problem

Cappanera and Scutellà (2011) introduce the problem of computing k balanced paths in a network, called the balanced path problem (BPP). The problem is to find k node-disjoint paths on a weighted network. They consider two main versions of BPP arising in many applications such as transportation and telecommunication networks. The single commodity version of the problem does not discriminate between paths. k node-disjoint paths emanating from k distinct source nodes and ending at k distinct sink nodes (considering all available source and sink nodes) are computed while the objective is to minimize the difference between the longest path and the shortest path. This version finds anonymous rosters in crew and personnel scheduling applications. When the problem is solved on a network representing the associated scheduling problem in a similar way, multi commodity version of the problem considers finding k node-disjoint paths using k source-sink pairs and generates personalized rosters. For the proposed mathematical formulation of the multi commodity version of BPP, they consider a network $G = (V, E)$ where there are k source-sink pairs as (s_h, t_h). x_{ij}^h denotes the 0-1 unit flow on arc (i, j) while Z_{max} and Z_{min} denote costs of the balanced longest and the shortest paths. The resulting mathematical formulation of the corresponding network problem is:

$$\text{min} \qquad Z_{max} - Z_{min} \tag{1}$$

$$\text{subject to} \qquad \sum_{(s_h,i)\in E} x_{s_h i}^h = 1 \qquad h \in \{1, \dots, k\} \tag{2}$$

$$\sum_{(i,t_h)\in E} x_{it_h}^h = 1 \qquad h \in \{1, \dots, k\} \tag{3}$$

$$\sum_{(j,i)\in E} x_{ji}^h - \sum_{(i,j)\in E} x_{ij}^h = 0 \qquad h \in \{1, \dots, k\}, \forall i \in V\{s_h, t_h\} \tag{4}$$

$$\sum_{(i,j)\in E} \sum_{h} x_{ij}^h \leq 1 \qquad \forall i \in V \tag{5}$$

$$u_i - u_j + n x_{ij}^h \leq n - 1 \qquad \forall (i, j) \in E, h \in \{1, \dots, k\}, i, j \notin \{s_h, t_h\} \tag{6}$$

$$\sum_{(i,j)\in E} c_{ij} x_{ij}^h \leq Z_{max} \qquad h \in \{1, \dots, k\} \tag{7}$$

$$\sum_{(i,j)\in E} c_{ij} x_{ij}^h \geq Z_{min} \qquad h \in \{1, \dots, k\} \tag{8}$$

$$x_{ij}^h \in \{0,1\} \qquad \forall (i, j) \in E, h \in \{1, \dots, k\} \tag{9}$$

$$1 \leq u_i \leq n \qquad \forall i \in V \tag{10}$$

Objective function (1) minimizes the difference between the paths with maximum cost and the path with minimum cost where cost of a path is the summation of the cost of the arcs on the path. Constraints (2)-(4) are traditional flow conservation constraints finding k paths from source nodes to sink nodes with zero flow balance for all nodes except source nodes and sink nodes. Constraints (5) ensure that paths are node-disjoint. Sub-tour elimination constraints are represented by constraints (6) where u_i is associated with node i can be interpreted as the position of node i in a tour. Constraints (7) and (8) keep costs of k paths in the range of optimal shortest and longest paths. Constraints (9) and (10) are domain constraints.

3.2. Network Representation

Şahin and Yüceoğlu (2011) present a time-space network for CCPP. It is possible to adapt this network representation for any transportation company considering different rules and policies of the company. In railways, there are one or more main crew regions which manage all operations of the region. Each region consists of a home station as its main station where all the crew related operations are managed. In addition, there are some away stations. Crew trips start from the home station (an away station) and end at an away (the home station). These trips are listed in a predetermined list of train duties. Other types of duties include station duties which are required to cover for absent crew of a train duty in case of emergency. In addition, it is possible to transfer a crew member without assigning any duty from one station to another in order to cover a duty starting at the destination station. Transferring crew members to another location (on trains covered by other crew members) is called deadheading. Deadheading plays an important role in covering duties with minimum number of crew members. A crew member can be transferred from an away station to the home station or vice versa. A deadheading from an away station to home station will result in accumulating crew members at home, which is useful when crew members are required for home to away duties. On the other hand, a deadheading from home station to an away station allows covering a duty starting from away station when there is no other crew member available at the away station to cover the duty.

The network representation in Şahin and Yüceoğlu (2011) also take into account a set of rules that are in practice determined by labor unions and imposed by the policy of the company. Some examples of these rules are as follows:

- There are predefined time periods to report for a train duty earlier than the departure of the train and to finish the duty later than the arrival of the train for crew members. These time windows, respectively called on-duty and off-duty times, are used for filling paperwork and debriefs on the trip.
- The maximum and the minimum rest time period between two consecutive trips for any crew member are predetermined. Crew members are subject to a rest time following a duty and prior to performing next duty (or deadheading). These rest times can differ for home and away stations.
- If the duration of a duty exceeds a predetermined length, it is required that at least two crew member are assigned to the duty.

Figure 1 shows the network representation designed by Şahin and Yüceoğlu (2011) for the Turkish State Railways example.

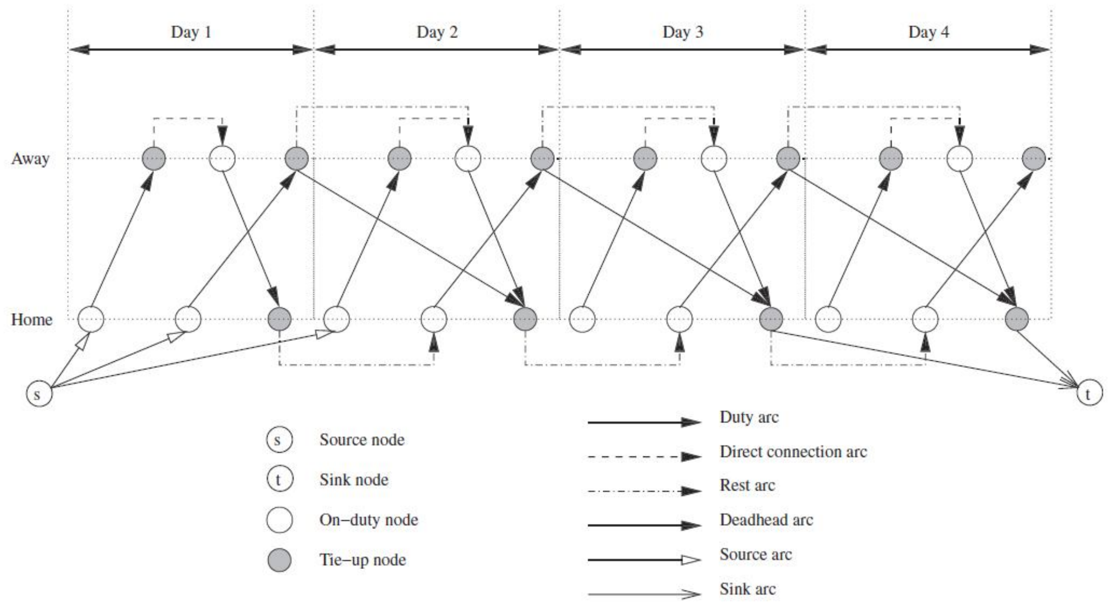


Figure 1 Network Representation for Turkish State Railways example

In the space-time network representations also suggested by Şahin and Yüceoğlu (2011), nodes contain time and location information and represent the beginning and ending of events. On-duty nodes denote the beginning time and location of a duty; tie-up nodes denote the end time and location. A source node is the origin of all crew members at the home station at the beginning of the planning time horizon. A sink node is the final destination of all crew representing the home station at the end of the planning horizon. The set of arcs includes six types:

- Source arcs emanate from the source node and enter the on-duty nodes at home location.
- Sink arcs emanating from tie-up nodes and ending at the sink node send all crew back to home station at the end of planning horizon.
- Duty arcs emanating from an on-duty node and entering a tie-up node represent duties; flow on duty arc represents the coverage of the duty. Each duty arc has a lower bound respecting minimum required number of crew members to cover the duty.
- Rest arcs have been used to represent rest periods which connect a tie-up node to an on-duty node at the same location.
- Direct arcs are used to connect two successive duties which have a total time duration less than a predefined time period. These arcs represent the coverage of an excess duty by a crew member. An excess duty covers the first duty, the waiting period between the two duties, and the second duty.
- Deadheading arc from an away tie-up node to a home tie-up node is used to transfer a crew member from the away station to the home station. Moreover, home to away deadheading is represented using duty arcs from home to away nodes, where over loading a duty arc means a deadheading from home to away.

On this space-time network, a source-sink path is composed of consecutive arcs which represent duties, rest periods and deadheading that correspond to a feasible schedule for a crew member from the beginning of the planning horizon until the end. As a result, each source-sink path would correspond to a feasible crew schedule. Şahin and Yüceoğlu (2011) propose a network flow problem on this network in order to find paths corresponding to such feasible crew schedules while the necessity for covering the duties are handled by appropriate lower bounds on the arcs representing different types of duties. Since they consider CCPP, their objective function minimizes the number of source-sink paths.

For the network flow formulation of CCPP they consider a network $G = (N, A)$ with a source node s and a sink node t where decision variable x_a denotes the amount of flow on arc $a \in A$. The set of duty arcs in network G is denoted by A_d. A^+/A^- denotes set of outgoing/incoming arcs at node i. Number of crew members required to cover the duty represented by arc a is demonstrated by c_a. The mathematical programming formulation for their network flow model is:

$$\min \quad \sum_{a \in A_{s^+}} x_a \tag{11}$$

$$\text{subject to} \quad \sum_{a \in A_{s^+}} x_a = \sum_{a \in A_{t^-}} x_a \tag{12}$$

$$\sum_{a \in A_{i^-}} x_a = \sum_{a \in A_{i^+}} x_a \qquad \forall i \in N \backslash \{s, t\} \tag{13}$$

$$x_a \geq c_a \qquad \forall a \in A_d \tag{14}$$

$$x_a \in \mathbb{Z}_+ \qquad \forall a \in A \tag{15}$$

The objective function (11) minimizes the flow emanating from the source node s which corresponds to the number of crew members required in the planning horizon. Constraint (12) is the flow balance constraint between the source node s and the sink node t, which ensures that the flow emanating from the source node is equal to the flow entering the sink node. We have the flow balance constraint of other nodes in (13). Constraint (14) is duty coverage constraint, which ensures for a duty arc the flow amount is at least as much as the number of required crew members, C_a. The integrality constraints on the variables are given in (15).

A solution to the problem in (11)–(15) is composed of a set of $s - t$ paths. The number of $s - t$ paths in the (optimal) solution is equal to the (optimal/minimum) number of crew members required. In essence, one could identify individual $s - t$ paths in a solution via post-processing with a depth-first search algorithm applied only on the arcs with nonnegative flow in the solution. Then, from the solution of the minimum flow problem, we may obtain a feasible assignment of duties to potential crew schedules with minimum number of crew members.

3.3. Workload Balancing in Crew Scheduling

A straightforward attempt to formulate the network flow model for WBP based on the network representation in Şahin and Yüceoğlu (2011) would modify the formulation (11)-(15) by updating the objective function with (1) and adding the constraints (7) and (8) from the BPP formulation. However, this straightforward approach seems to have some problems as we discuss next. Based on the network representation in Şahin and Yüceoğlu (2011), we develop a mathematical programming formulation of WBP inspired from the formulation of BPP in Cappanera and Scutellà (2011). Due to some particular characteristics and the nature of the problem, the new formulation for WBP is indeed quite different from that of BPP. The node-disjoint constraints in BPP ensure that paths are dissimilar from each other. Without this constraint the problem would result in k equivalent paths and the objective function would be equal to zero.

In the network flow model for CCPP, paths are not necessarily node-disjoint. As a result, the straightforward adaptation approach would yield artificially inflated workload for some crew so that the difference between the maximum and the minimum is smaller. This is possible particularly due to the feasibility of freely adding unnecessary deadheading to the crew schedules. In essence, the total cost of all paths (corresponding to the total workload in crew schedules) is given and implicitly fixed in BPP which is imposed by the node-disjointness of the paths. However, the opportunity for adding unnecessary workload such as crew deadheading and the absence of disjointness constraints make it possible to create extra workload and assign it to the crew in order to attain a more balanced workload at the expense of extra crew cost for unnecessarily inflated workloads.

Based on the network representation $G = (N, A)$, WBP finds a set of k balanced paths where k is the optimal solution of CCPP. In the network flow formulation of WBP, $x_{ij}^{h} = 1$ if arc (i, j) is included in the feasible path h; otherwise, $x_{ij}^{h} = 0$. When (i, j) is included in a path, it means that it is covered by the schedule represented by the path. W_{ij} denotes the workload of arc (i,j); it is zero for all arcs other than duty and deadhead arcs. For a duty arc (i,j), c_{ij} is the number of crew members required to cover the duty. ϵ is an extremely small positive quantity. Then, the network flow formulation for WBP becomes:

min $\quad Z_{max} - Z_{min}$ (16)

subject to

$$\sum_{(s,i)\in A} x_{si}^h = 1 \qquad h \in \{1,\dots,k\} \tag{17}$$

$$\sum_{(i,t)\in A} x_{it}^h = 1 \qquad h \in \{1,\dots,k\} \tag{18}$$

$$\sum_{(j,i)\in A} x_{ji}^h - \sum_{(i,j)\in A} x_{ij}^h = 0 \qquad h \in \{1,\dots,k\}, \forall i \in N\{s,t\} \tag{19}$$

$$\sum_{h} x_{ij}^h \geq c_{ij} \qquad \forall (i,j) \in A \tag{20}$$

$$\sum_{(i,j)\in A} W_{ij} x_{ij}^h \leq Z_{max} \qquad h \in \{1,\dots,k\} \tag{21}$$

$$\sum_{(i,j)\in A} W_{ij} x_{ij}^h \geq Z_{min} \qquad h \in \{1,\dots,k\} \tag{22}$$

$$\sum_{h} \sum_{(i,j)\in A} W_{ij} x_{ij}^h \leq W_T + \epsilon \qquad \forall (i,j) \in A, h \in \{1,\dots,k\} \tag{23}$$

$$x_{ij}^h \geq 0 \qquad h \in \{1,\dots,k\}, \forall (i,j) \in A \tag{24}$$

The objective function (16) minimizes the workload difference between the two crew schedules (i.e. feasible paths). Constraints (17), (18) and (19) are flow conservation constraints which assure that, for each path h, a unitary flow is pushed from the source node to the sink node, representing the engagement of only one crew with the feasible schedule represented by the path. Constraint (20) assures that all duties are covered. Constraints (21) and (22) ensure that all paths have a workload within the range of the minimum and the maximum workload, and also determine the value of the workload for the maximum-workload schedule/path and the minimum-workload schedule/path, respectively.

Constraint (23) assures that the total workload of all paths does not exceed the minimum total workload and constraint (24) is the domain constraint for the decision variables.

Without preventing excessive use of deadheading, the WBP in crew scheduling results in assigning extra duties to crew. We should clearly state that constraint (23) is the resolution of the problem with the straightforward adaptation of the network flow model for CCPP; it avoids the addition of unnecessary workload through deadheading by limiting the total workload that can be assigned to crew. In essence, W_T can be calculated as the optimal objective function value of a variant of CCPP where the objective function minimizes the total workload rather than the number of crew members. The corresponding mathematical formulation is:

$$W_T = \min \quad \sum_{h} \sum_{(i,j)\in A} W_{ij} x_{ij}^{h} \qquad (25)$$

$$\text{subject to} \quad (17) - (20) \; and \; (24)$$

The proposed network flow formulation (16)-(24) finds an optimal solution for WBP in crew scheduling. In order to decrease the computational effort required to solve the problem some improvements shall be done by narrowing down the feasible region of the problem. In addition to a feasible upper bound on the total workload W_T, we also impose a lower bound as $\sum_h \sum_{(i,j)\in A} W_{ij} x_{ij}^{h} \geq W_D + \epsilon$ where the lower bound W_D can be found as the summation of workload of all (train and station) duties with no deadheading to be covered within the planning horizon.

Chapter 4

Exact Algorithm

In this section, we introduce an exact algorithm to solve WBP. This algorithm avoids solving the mathematical programming problem formulation; rather, it obtains the optimal solution using a binary search method. In essence, we first determine the interval where the optimal objective function value (i.e. the difference between the path with maximum cost and the path with minimum cost) lies in; then, we explore this interval to find out where a feasible solution with the minimum difference exists.

4.1. Cost Expanded Network

For any directed and weighted network $G = (N, A)$ with a source node $s \in N$, the cost expanded network $G_U = (N_U, A_U)$ is defined as an un-weighted directed network containing the source node s of network G and several copies of all other nodes. Each node $i \in N$ is represented by at most $U + 1$ copies in G_U where U is the cost of the longest path in G. On the cost expanded network G_U, $i^p \in N_U$ represents a copy of $i \in N$ while p is the cost of s directed path from s to i. $s \in N$ is represented by only one copy as s^0 in G_U. A weighted arc $(i, j) \in A$ with cost c is represented by multiple un-weighted arcs $(i^p, j^{p+c}) \in A_U$ for all possible values of p. If there exist a sink node t, the sink node $t \in A$ is represented by $t^p \in N_U$, where $p \in \{0,1, \dots, U\}$. Any t^p shows a directed source-sink path with cost of p on the original network. Following is an algorithm for generating the cost expanded network for network G:

```
**Generate Cost Expanded Network** ($G$)**:**
**Add** the source node $s^0$ to $G_U$
**Topological Order** ($G$);
**For** $i = 0$; $i <= Max\_Top\_Order$; $i++$ **do**
    **For** each arc $(i, j)$ with cost $c_{ij}$ **do**
        **For** each $i^p$ in $G_U$ **do**
            q=$p$+$c_{ij}$;
            **Add** node $j^q$ to $G_U$;
            **Add** arc $(i^p, j^q)$ to $G_U$;
**Return** $G_U$;
```

Representing each node of the original network with at most $U + 1$ copies in the cost expanded network, the set of nodes of the cost expanded network N_U, will have $O(Un)$ nodes. Similarly, the cost expanded network contains $O(Um)$ arcs. Considering that U is the cost of the longest path in the network G, the size of the nodes and the arcs of the cost expanded network can be very large.

4.2. Binary Search Algorithm

We develop a binary search algorithm to solve WBP. It is clear that $[0, U]$ contains the optimal objective function value. The binary search algorithm (BSA) starts with $\alpha = U/2$ and checks if there are k paths satisfying constraints (2)-(5), (8) and (9) where the difference between shortest and longest path is equal to α. If the result is positive, then the search continues on the interval $[0, \alpha]$; if not, the interval changes to $[\alpha, U]$. The algorithm continues in this fashion until the right hand side (RHS) and the left hand side (LHS) of the search interval is equal to each other. Figure 2 shows a flow chart of BSA.

The proposed network G, the minimum total workload W_T and the cost of the longest path U are inputs of BSA. The algorithm initials LHS of the search interval a zero and the RHS of the search interval as U. Then, BSA checks if there is a feasible subset selection of schedules with $\alpha = (LHS + RHS)/2$. If the result is positive RHS changes to α; otherwise, LHS changes to α. The algorithm continues until RHS and LHS are equivalent.

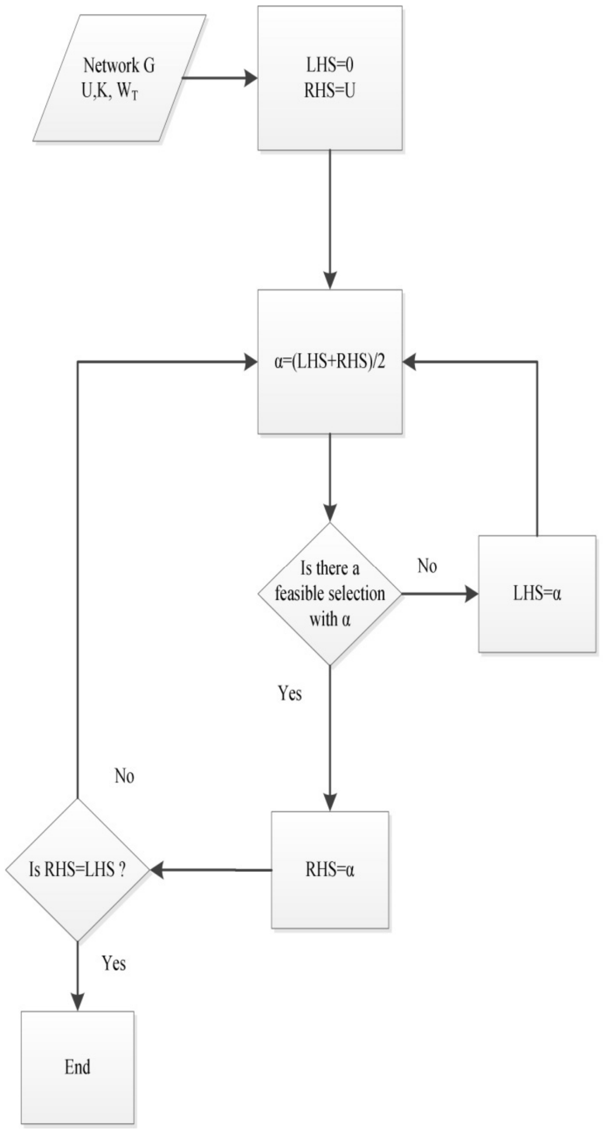


Figure 2 Binary Search Algorithm

4.3. Feasibility Check Procedure

During one iteration of BSA, the major task is to figure out if the problem has a feasible solution given a difference value (α) corresponding to the objective function value of WBP. Therefore, in order to check if there are k paths satisfying constraints (17)-(20) and (23)-(24) where the difference between shortest and longest path is equal to α, the cost expanded network G_U is used. For each value of α, a family of sub-graphs of G_U is employed. Recall that G_U contains a set of sink nodes $T = \{t^q, q \in \{0, \dots, U\}\}$. For a given value α, one should seek feasibility in a series of problems where the network includes a subset of the sink nodes $T_q^\alpha = \{t^q; q \in \{q', \dots, q' + \alpha\}\}$. In essence, if there are k paths on T_q^α covering the required duty arcs, corresponding WBP instance has a solution where the difference of maximum workload and minimum workload is α.

Let each instance of such sub-graphs be denoted by $G_\alpha^q = (N_\alpha^q, A_\alpha^q)$ for $q \in \{q_0, \dots, q_U\}$. For a fixed α value, if at least one of these sub-graphs has k paths where all duties are covered and total workload does not exceed the minimum total workload, W_T, then α is an upper bound for the optimal value of WBP.

There are at most $U + 1$ subgraphs for each value of α. Considering the minimum total workload constraint, sub-graphs G_α^q with smaller values of q are most likely to have k paths where coverage and minimum total workload constraints are satisfied. It is sufficient to start with $G_\alpha^{q_0}$ and check the feasibility for this sub-graph. If the result is positive, the iteration for α is terminated and the feasibility check procedure ends with a positive result. If the result is negative, the iteration continues to check for larger values of q in the same fashion. For each sub-graph G_α^q, a feasibility problem is solved. We should check if the sub-graph contains k paths from source node s to sink nodes with aforementioned constraints. The exact formulation for this feasibility problem $[F1]$ is as follows:

$$\sum_{(s,i) \in A_\alpha^q} x_{si} = k \qquad (26)$$

$$\sum_{t\in\{t^q,\dots,t^{q+\alpha}\}} \sum_{(i,t)\in A_\alpha^q} x_{it} = k \tag{27}$$

$$\sum_{(j,i)\in A_\alpha^q} x_{ji} - \sum_{(i,j)\in A_\alpha^q} x_{ij} = 0 \qquad \forall i \in N_\alpha^q \backslash \{s, t^q, \dots, t^{q+\alpha}\} \tag{28}$$

$$\sum_{(i,j)\in R_{kl}} x_{ij} \geq C_{kl} \qquad \forall (k,l) \in A \tag{29}$$

$$\sum_{t\in\{t^q,\dots,t^{q+\alpha}\}} W_t \sum_{(i,t)\in A_\alpha^q} x_{it} \leq W_T + \epsilon \tag{30}$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in A_\alpha^q \tag{31}$$

Constraints (26), (27) and (28) are flow conservation constraints which assure that k unit flows are pushed from the source node to sink nodes. Constraint (29) assures that, all duties are covered. Note that for each arc there are several copies in sub-graph G_α^q shown by R_{kl}, $(k,l) \in A$. Constraint (30) assures that the total workload of all paths does not exceed the minimum total workload W_t. Constraint (31) is the domain constraint for the decision variables.

The flow chart of BSA using the feasibility check procedure is shown in Figure 3. Note that for fixed value of α, the algorithm checks if $[F1]$ is true at least on one of sub-graphs G_α^q starting from $q = q_0$. If the iteration terminates with negative result, the algorithm continues with changing LHS to α; otherwise, RHS changes to α.

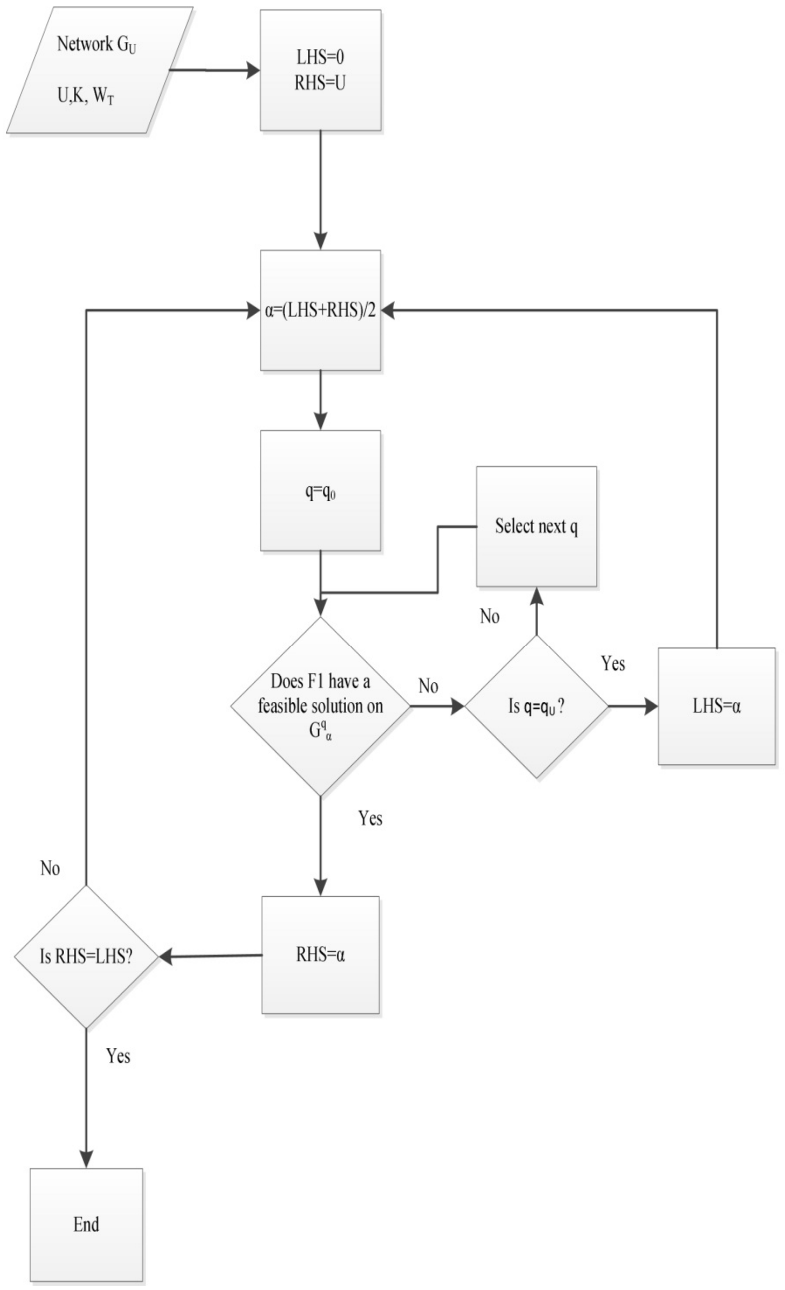


Figure 3 Revised BSA

4.4. Binary Search on Improved Interval

A binary search in interval $[0, U]$ includes performing $log(U)$ feasibility check iterations. By decreasing the distance between the two sides of the interval, the number of iterations of the algorithm is expected to decrease. U, on the right hand side of the interval, represents the case when the maximum workload is U, i.e. the length of the longest source-sink path, and the minimum workload is 0. Instead, a lower bound for the right hand side such as $Z_{max}^{UB} - Z_{min}^{LB} \leq U$ would narrow down the search interval from the right hand side where Z_{max}^{UB} denotes an upper bound for the maximum workload (i.e. the length of the longest path) and Z_{min}^{LB}denotes a lower bound for the minimum workload (i.e. the length of the shortest path).

In this respect, if

> Z_{max}^{UB} is an upper bound for the workload of the schedule with maximum workload in an optimal solution of WBP (Z_{max}^{*}) , and
> Z_{min}^{LB} is a lower bound for the workload of the schedule with minimum workload in an optimal solution of WBP (Z_{min}^{*}),

then, the difference between Z_{max}^{UB} and Z_{min}^{LB} is an upper bound for the optimal value of optimal objective function value (equation (16)), i.e. $Z_{max}^{*} - Z_{min}^{*} \leq Z_{max}^{UB} - Z_{min}^{LB}$.

In order to find Z_{max}^{UB} we can modify the mathematical formulation of WBP. Extra constraints are needed to prevent an infinite value for the upper bound. Z_{max}^{UB} is the optimal value of the following problem [P_{max}^{UB}]:

Max $\qquad \bar{Z}_{max}$ (32)

subject to $\qquad (17) - (20)\ \ and\ (23) - (24)$

$$\sum_{(i,j)\in A} W_{ij}x_{ij}^{h} + M(1 - y^{h}) \geq \bar{Z}_{max} \qquad h \in \{1, \dots, k\} \tag{33}$$

$$\sum_{(i,j)\in A} W_{ij}x_{ij}^{h} \leq \bar{Z}_{max} \qquad h \in \{1, \dots, k\} \tag{34}$$

$$\sum_{h} y^{h} \geq 1 \tag{35}$$

$$y^{h} \in \{0,1\} \qquad h \in \{1, \dots, k\} \tag{36}$$

where $y^{h} = 1$ if the corresponding path is the feasible schedule with the maximum possible workload; $y^{h} = 0$, otherwise. The objective function (32) maximizes the workload of the feasible schedule with the maximum workload. Constraints (33), (34) and (35) ensure that there is at least one schedule with workload amount of $\bar{Z}_{max}$. Constraint (36) is the binary domain constraint.

Similarly, by modifying the mathematical formulation of WBP, a mathematical formulation to find Z_{min}^{LB} can be developed. The optimal value of the following problem ($[P_{min}^{LB}]$) is equal to Z_{min}^{LB}:

$$\text{Min} \quad \bar{Z}_{min} \tag{37}$$

$$\text{subject to} \quad (17)-(20) \; and \; (23)-(24)$$

$$\sum_{(i,j)\in A} W_{ij}x_{ij}^{h} + M(1-y^{h}) \geq \bar{Z}_{min} \qquad h \in \{1,\dots,k\} \tag{38}$$

$$\sum_{(i,j)\in A} W_{ij}x_{ij}^{h} \geq \bar{Z}_{min} \qquad h \in \{1,\dots,k\} \tag{39}$$

$$\sum_{h} y^{h} \geq 1 \tag{40}$$

$$y^{h} \in \{0,1\} \qquad h \in \{1,\dots,k\} \tag{41}$$

where $y^{h} = 1$ if the corresponding path is the feasible schedule with the minimum possible workload; $y^{h} = 0$, otherwise. The objective function (37) minimizes workload of the schedule with the minimum possible workload. Similar to $[P_{max}^{UB}]$, Constraints (38), (39) and (40) ensure that there is at least one path with workload amount of $\bar{Z}_{min}$. Constraint (41) is the binary domain constraint.

In order to narrow down the search interval from the left hand side, instead of using zero, a better lower bound for the optimal value of equation (16) can be found. If Z_{max}^{LB} is a lower bound for Z_{max}^{*} and Z_{min}^{UB} is an upper bound for Z_{min}^{*}, $Z_{max}^{LB} - Z_{min}^{UB}$ is a lower bound for the optimal value of equation (16), i.e. $Z_{max}^{LB} - Z_{min}^{UB} \leq Z_{max}^{*} - Z_{min}^{*}$. In this respect, we develop relaxations of the network flow formulation for WBP.

We first consider a relaxation of the feasible region of (16)-(24) without constraint (22). Since (16) implicitly minimizes Z_{max}, setting the objective function to minimization of Z_{max} only produces a lower bound. Therefore, we use the following problem formulation ($[P_{max}^{LB}]$) to find Z_{max}^{LB}:

min $\bar{Z}_{max}$ (42)

subject to $(17)-(21)\ and\ (23)-(24)$

In a similar fashion, deleting constraint (21) from formulation (16)-(24) and setting the objective function to maximize Z_{min}, an upper bound for $Z^{*}_{\min}$ can be obtained. Clearly, the feasible region of the new problem $[P^{UB}_{min}]$ is a relaxation of (17)-(24). Therefore, Z^{UB}_{min} is the optimal objective function value of the following is the mathematical formulation $[P^{UB}_{min}]$:

max $\bar{Z}_{min}$ (43)

subject to $(17)-(20), (22)\ and\ (23)-(24)$

$PROPERTY\ 1$. A binary search on $[0, U]$ for the optimal solution of WBP is equivalent to a binary search on $[(Z^{LB}_{max} - Z^{UB}_{min}), (Z^{UB}_{max} - Z^{LB}_{min})]$.

As a result of Property 1, BSA will speed up and the number of iterations will be equal to $\log\,[(Z^{UB}_{max} - Z^{LB}_{min}) - (Z^{LB}_{max} - Z^{UB}_{min})]$.

4.5. Improvements on the Feasibility Check Procedure

During the feasibility check procedure of BSA, we observe that there is no need to check all sub-graphs G^{q}_{α} for a fixed value of α. Constraint (30) imposes that the total workload cannot exceed W_T. In this respect, if the sum of the cost labels of the first k sink nodes (corresponding to the total workload of first k schedules) exceeds the minimum total workload W_T, then the result of the feasibility problem $[F1]$ for this sub-graph will be negative.

$PROPERTY$ 2. For a sub-graph G_{α}^{q}, if the sum of the cost labels of the first k sink nodes exceeds the minimum total workload W_T, then none of the sub-graphs $G_{\alpha}^{q'}$ where $q' \geq q$ and $q' \leq U$ will not satisfy the feasibility problem $[F1]$.

We may also call Property 2 as the *stopping sink node* property as it specifies where to stop the search for a given value of α. With the list of sink nodes of network G_U sorted based on cost as input, an algorithm to find the *stopping sink node* can be summarized as follows:

```
**Find Stopping Node** ($S$):
$t^{q_{Stopping}} = 0$
Stop=False;
$N = \emptyset$;
**While** Stop! =True **do**
    **For** all $t^q \in S$ **do**
        **Select** first $k$ nodes and add to list $N$
        **If** sum of elements ($N$)> $W_T$ **do**
            Stop=True;
            $t^{q_{Stopping}} = t^q$;
**Return** $t^{q_{Stopping}}$;
```

For a given value of α, we may define the set of G_{α}^{q} where $q = q_1, \dots, q_{Stopping}$. To avoid checking all of these sub-graphs for a specific value of α one by one, all of them can be combined in a new sub-graph and execute the feasibility check procedure for a combined sub-graph called $G_{\alpha} = (N_{\alpha}, A_{\alpha})$. It would then suffice to check if G_{α} with sink nodes $\{t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\}$ contains k paths from the source node to sink nodes where all duties are covered and the difference between the length of each pair of paths is at most α. For this purpose, we develop the following mathematical programming formulation for this new feasibility problem $[F2]$:

$$\sum_{(s,i)\in A_\alpha} x_{si} = k \tag{44}$$

$$\sum_{t\in\{t^{q_1},\dots,t^{q_{Stopping}+\alpha}\}} \sum_{(i,t)\in A_\alpha} x_{it} = k \tag{45}$$

$$\sum_{(j,i)\in A_\alpha} x_{ji} - \sum_{(i,j)\in A_\alpha} x_{ij} = 0 \qquad \forall i \in N_\alpha \backslash \{s, t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\} \tag{46}$$

$$\sum_{(i,j)\in R_{kl}} x_{ij} \geq C_{kl} \qquad \forall (k,l) \in A \tag{47}$$

$$\sum_{t\in\{t^{q_1},\dots,t^{q_{Stopping}+\alpha}\}} W_t \sum_{(i,t)\in A_\alpha} x_{it} \leq W_T + \epsilon \tag{48}$$

$$M * \left(\sum_{(i,j)\in A_\alpha} x_{ij} \right) - y_t \geq 0 \qquad \forall t \in \{t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\} \tag{49}$$

$$\varepsilon * \left(\sum_{(i,j)\in A_\alpha} x_{ij} \right) - y_t \leq 0 \qquad \forall t \in \{t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\} \tag{50}$$

$$y_{t_1} W_{t_1} - y_{t_2} W_{t_2} - M\left(2 - y_{t_1} - y_{t_2}\right) \leq \alpha \qquad \forall t_1, t_2 \in \{t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\} \tag{51}$$

$$y_{t_1} W_{t_1} - y_{t_2} W_{t_2} + M\left(2 - y_{t_1} - y_{t_2}\right) \geq -\alpha \qquad \forall t_1, t_2 \in \{t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\} \tag{52}$$

$$y_t \in \{0,1\} \qquad \forall t \in \{t^{q_1}, \dots, t^{q_{Stopping}+\alpha}\} \qquad (53)$$

$$x_{ij} \geq 0 \qquad \forall (i,j) \in A_\alpha \qquad (54)$$

Constraints (44)-(48) are similar to constraints (26)-(30) in $[F1]$. Constraints (49), (50) and (53) ensure that $y_t = 1$ if there is incoming flow at sink node t; otherwise, $y_t = 0$. Constraints (51) and (52) assure that the difference between workload of any pair of schedules is equal to α. The result of (44)-(54) is positive, if G_α contains k paths from the source node to sink nodes where all duties are covered and the difference between the length of each pair of paths is at most α.

4.6. Properties of the Optimal Solution

In order to reduce the computational burden at each iteration of BSA, we explore other useful properties of the optimal solution. If the lower bound and the upper bound for $Z^*_{\min}$ are equal, and in a similar way if the lower bound and the upper bound for $Z^*_{\max}$ are equal, the following property is useful:

$PROPERTY$ 3.

a) *If* $Z^{LB}_{min} = Z^{UB}_{min}$ *then* $Z^*_{\min} = Z^{LB}_{min} = Z^{UB}_{min}$ *and the optimal solution of the workload balancing problem is equal to* $Z^{LB}_{max} - Z^{LB}_{min}$.

b) *If* $Z^{LB}_{max} = Z^{UB}_{max}$ *then* $Z^*_{\max} = Z^{LB}_{max} = Z^{UB}_{max}$ *and the optimal solution of the workload balancing problem is equal to* $Z^{UB}_{max} - Z^{UB}_{min}$.

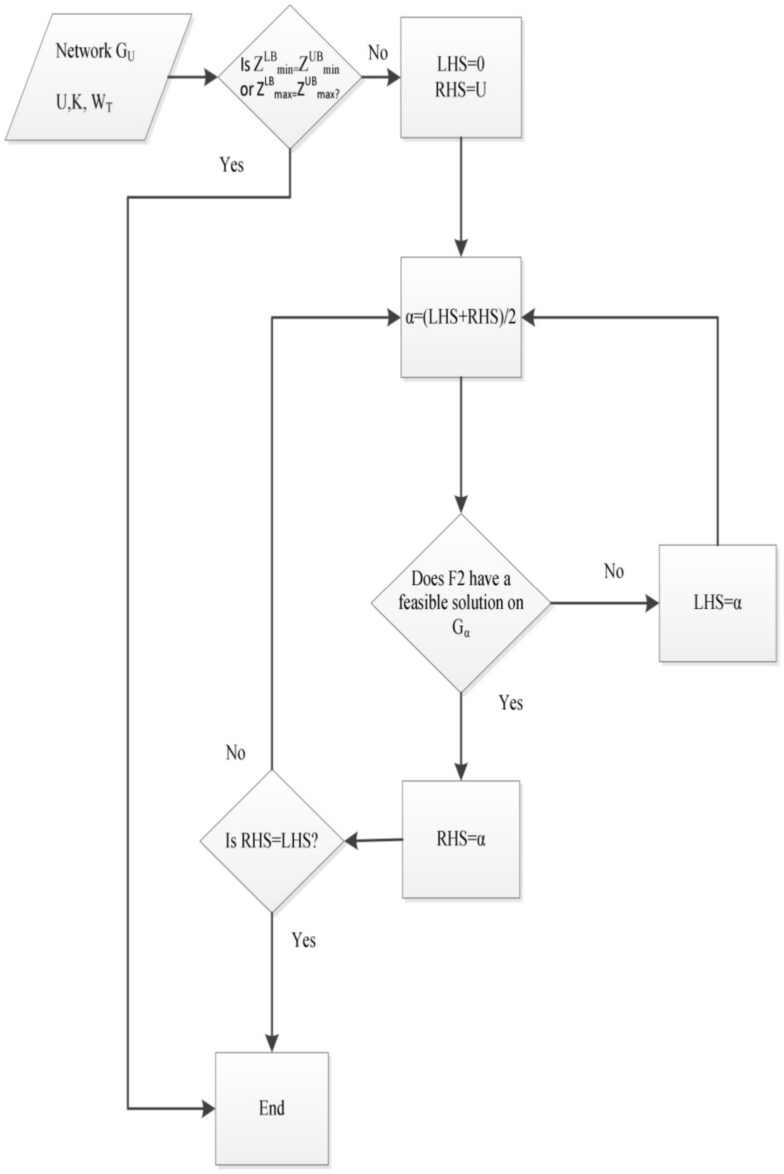
To prove Property 3 part (a), let's suppose that $Z^{LB}_{max} - Z^{LB}_{min} \neq Z^*_{max} - Z^*_{min} \Rightarrow Z^{LB}_{max} \neq Z^*_{max}$ then:

1) $Z^{LB}_{max} > Z^*_{max}$ which is a contradiction.

2) $Z^{LB}_{max} < Z^*_{max} \Rightarrow$ Optimal solution of $[P^{LB}_{max}]$ $x^*_{h,LB,max} < Z^*_{min}$ at least for one value of $h \Rightarrow x^*_{h,LB,max} < Z^{LB}_{min}$ at least for one value of $h \Rightarrow x^*_{LB,max}$ is an optimal solution of $[P^{LB}_{min}]$ which is a contradiction.

From (1) and (2), one may conclude that $Z_{max}^{LB} = Z_{max}^{*}$ (and $Z_{max}^{LB} - Z_{min}^{LB} = Z_{max}^{*} - Z_{min}^{*}$). A similar proof can be presented for Property 3 part (b).

Using Property 3, Figure 4 shows the flow chart of the exact cost extended network based BSA. At each iteration for a fixed value of α, $[F2]$ is checked for G_{α}. If the iteration terminates with negative result, the algorithm continues with changing LHS to α; otherwise, RHS changes to α.



Network G_U
U, K, W_T
Is $Z^{LB}_{min} = Z^{UB}_{min}$ or $Z^{LB}_{max} = Z^{UB}_{max}$?
No
Yes
LHS=0
RHS=U
α=(LHS+RHS)/2
Does F2 have a feasible solution on G_α
No
LHS=α
Yes
RHS=α
Is RHS=LHS?
No
Yes
End

Figure 4 the exact cost extended network based exact BSA

The exact cost extended network based exact BSA can be summarized as follows:

```
**Input** $G, k, L, U, W_T$
**If** $Z_{min}^{LB} = Z_{min}^{UB}$ or $Z_{max}^{LB} = Z_{max}^{UB}$ **do**
    **Output** $Z_{max}^{LB} - Z_{min}^{UB}$;
$G_U$=**Generate Cost Expanded Network** $(G)$**;**
$S$=list of sink nodes of $G_U$ sorted based on cost;
$t^{q_{Stopping}}$ = **Find Stopping Node (**$S$): 
**Return Binary Search** $(L, U)$;
```

Where the *Binary Search* functions works as follows:

```
**Binary Search (**$Min, Max$**):**
**If** $Min == Max$ **Return** $Min$;
$\alpha = Min + (Max - Min)/2$;
$P$=First element of the sorted sink nodes;
**While** $p < t^{q_{Stopping}} + \alpha$ **do**
    Add $p$ and all the nodes connected to $p$ to the sub graph $G'$
**If** $G'$ contains all the duty arcs of network $G$ **do**
    **If** Check-Feasibility ($W\_T$ , $\delta$, $G'$)==True **do**
        **Binary Search** ($Min$, $\delta$);
        Break;
**Binary Search** ($\delta$, $Max$);
```

Chapter 5

Heuristic Methods

Heuristic ideas are introduced in this section. Multiple conventional heuristic algorithms are developed in order to find good feasible solutions and optimal solution when it is possible for the workload balancing problem.

5.1. Conventional Heuristics

Our conventional heuristic framework consists of a pool of different algorithms based on traditional neighborhood search ideas. The heuristic works iteratively and tires to improve the workload difference between the schedule with maximum workload and the schedule with minimum workload each iteration. From a feasible solution, a neighboring solution is obtained by exchanging some duties between a pair of schedules. An initial set of feasible schedules is required as input, which can be generated from an optimal or a feasible solution of the minimum capacity problem. These schedules are possibly unbalanced with respect to the difference between the maximum and the minimum workload.

At each iteration of the algorithm, a pair of schedules is selected. A neighborhood operator looks for a neighboring solution by exchanging a set of duties between these schedules. A neighboring solution is called improving if it decreases the workload gap between the selected pair. The algorithm works in the following manner:

```
Input initial set of feasible schedules
While there is an improvement do
```

```
        Select a pair of schedules;
        Perform the operator on the pair;
        Update set of schedules;
Output set of schedules;
```

Two main parameters of the heuristic are the operator and the selection criteria; we develop two operators and three different selection criteria.

5.1.1. Operators

The Crisscross operator works on a pair of schedules; it partitions each schedule into two parts in such a way that first part of the first schedule is continued by second part of the second schedule and vice versa. A feasible connection may be performed if the schedules are partitioned through partially or completely overlapping rest periods. Therefore the schedules remain feasible with respect to minimum and maximum home and away rest constraints. The operator starts by finding a feasible crisscross point in schedules. The feasible point is either a home tie-up node or an away tie-up node. To assure the feasibility of the crisscross operation, the algorithm checks the following:

- If both schedules are at the end of a home to away/away to home trip, then after performing the crisscross action, the rest period of both crew members should be between minimum and maximum home/away rest periods.
- If both schedules are at the end of a home to away then the location of the away stations in both should be the same.

For any pair of schedules there could be more than one, and indeed several possible crisscross operations. The feasibility of crisscross operation in both schedules is ensured while the difference between the workloads of schedules for a possible crisscross is calculated. If the workload gap is to decreases for a possible crisscross then the change is improving and the amount of the improvement is recorded. The operator checks all such improving changes in the pair and performs the crisscross with best improvement. If no such improvement is found, the crisscross operator ends without any change in the pair.

Figure 4 shows a pair of schedules demonstrated as paths on the network representation with a feasible crisscross operation. The new pair of schedules is shown in Figure 5 where schedules are generated by partitioning the schedules from rest periods (A,C) and (B,E) and replacing them with rest periods (A,E) and (B,C).

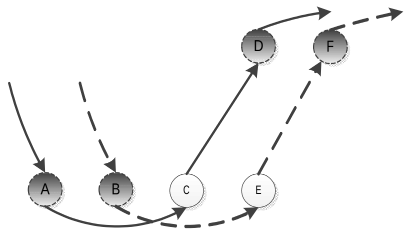


Figure 5 Pair of schedules before Crisscross

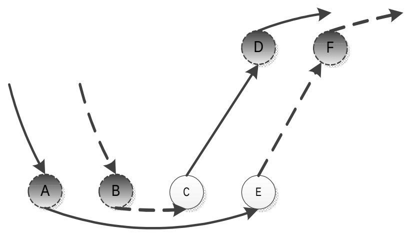


Figure 6 Pair of schedules After Crisscross

The Swap operator replaces a pair of consecutive duties together with the corresponding rest periods among the pair of schedules. Swap operation consists of a trip from home to an away station, an away rest period and a trip from the away station back to home station. For each schedule, the operator starts with the first duty in the first schedule of the pair and searches for a duty in the second schedule of the pair partially or completely overlapping with the first duty. The overlap depends on the home and away rest periods. If such a duty in the second schedule is not found the operator tries with another duty from the first schedule. As in the procedure for the crisscross operation, the swap operation with the best improvement is selected. If there is no such improvement in workload gap the operator terminates with initial pair of schedules kept without any change. Figure 6 shows a pair of schedules demonstrated as paths on the network representation with duties to be swapped. By replacing these duties, new schedules in Figure 7 are obtained.

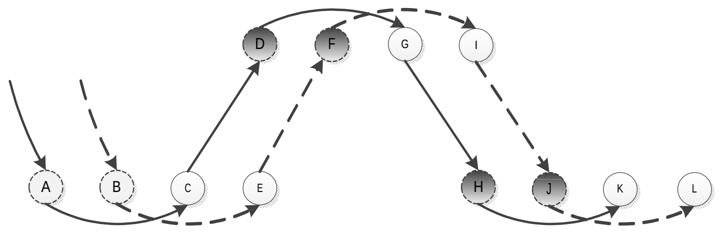


Figure 7 Pair of schedules before Swap

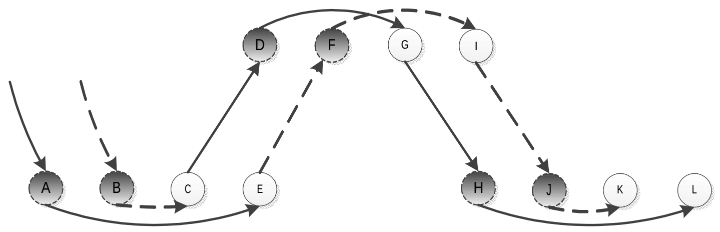


Figure 8 Pair of schedules after Swap

Both crisscross and Swap operators highly depend on the characteristics and parameters of the problem. It is possible to use these operators in any kind of transportation crew scheduling problem for which our network representation can be used. Yet, the performance of the operators may differ in different instances and none of each dominates the other. In order to obtain a better local improvement, it is possible to use the operator with a better improvement for any pair of schedules. For a selected pair, both operators are examined and the operator with the better improvement can be performed. A Hybrid Operator could be designed using this procedure.

5.1.2. Selection Criteria

Each iteration of the algorithm first selects a pair of schedules that might be changed at the end of the iteration. In order to select the pair to perform the operator alternative selection methods are used. In essence, a selection method is expected to find the pairs that are more likely to improve the workload gap. We propose three alternative selection methods.

In Fixed List selection method, a sorted list of k initial schedules in non-decreasing order with respect to workload amount is used. The method iterates as follows:

- At the first iteration, the first (minimum-workload) and the last (maximum workload) schedules are selected.
- The first schedule and $(k-1)^{th}$ schedule are selected.
- The pair selection proceeds from the bottom of the list until the first and the second schedules are selected. If no improvement is attained, the second schedule is paired with the last one.

The algorithm continues in this fashion until the first schedule of the selected pair is in the middle of the list.

Comparative Gap method selects pairs with respect to workload gap in any iteration. This method works in the following fashion:

- At the first iteration, the first and the last schedules are selected as they have the maximum workload gap.
- If the workload gap of the first and $(k-1)^{th}$ schedules is bigger than the workload gap of the second and the last schedules, they are selected. Next comparison should be made between workload gap of pair of the second and the last schedules and pair of the first and $(k-2)^{th}$ schedule.
- If the workload gap of the second and the last schedule is bigger than the workload gap of the first and $(k-1)^{th}$ schedules, they are selected. Next comparison should be made between workload gap of pair of the first and $(k-1)^{th}$ schedules and pair of the third and the last schedule.

The algorithm continues in this fashion until all pairs are examined.

Sorted Gap is an extension of the Comparative Gap method where pairs of schedules are listed in a non-increasing order according to the workload gap amount. The algorithm selects the first and the last schedule as they have the maximum gap. Then the selection process continues using the generated list.

Static and dynamic versions of the selection methods are designed. There are particular differences between Static and Dynamic versions of selection methods. In the static version, the process continues in the predetermined iterative manner. Whereas in the Dynamic version, if a operators manipulate the crew schedules, the crew selection procedure starts from the very beginning with new set of crew schedules.

A pool of conventional heuristic algorithms is designed using different combinations of the operators and selection methods.

Chapter 6

Computational Study

Our computational experience is conducted with two sets of problems in the context of transportation crew scheduling. The exact mathematical programming formulation (using CPLEX without additional cuts or inequalities) and BSA are implemented using C++ and CPLEX Concert Technology.

6.1. Crew Scheduling Instances from Literature

Beasley and Cao (1996) proposed a set of crew scheduling instances. Each instance consists of a depot and a set of duties to be covered by crew. Instances are represented by a duty network, where nodes represent duties. The length of each duty (the time duration required to perform the duty) is given. For any pair of duty nodes, there is a transition arc with an associated transition cost c if it is possible for a crew to perform these duties consecutively. The depot is shown using one source node and one sink node. Figure 8 shows the proposed network representation.

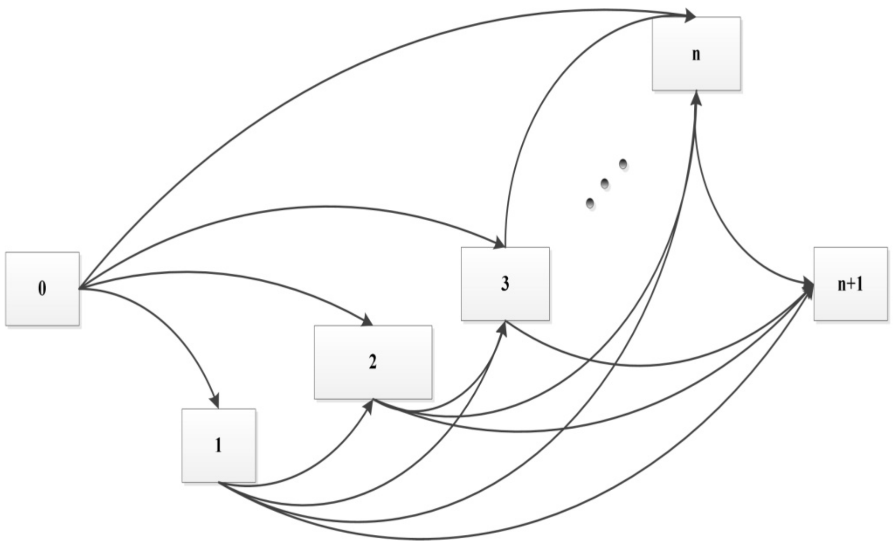


Figure 9 Network representation of crew scheduling instances in Beasley and Cao (1996)

We modify these instances adding the possibility of deadheading for crew members. Consequently, the proposed network representation is modified and deadheading arcs are added. According to Şahin and Yüceoğlu (2011), each duty node is converted to an on-duty node, tie-up node and a duty arc emanating from the on-duty node and ending at the tie-up node. For each transition arc (i, j) with cost of c in the original network, a rest arc from the associated tie-up node of the duty i to the associated on-duty node of duty j with the cost of c is added in the modified network. In addition, a deadheading is represented by an arc emanating from the associated tie-up node of duty i to the associated tie-up node of duty j with the cost of c. Figure 9 shows the modified network.

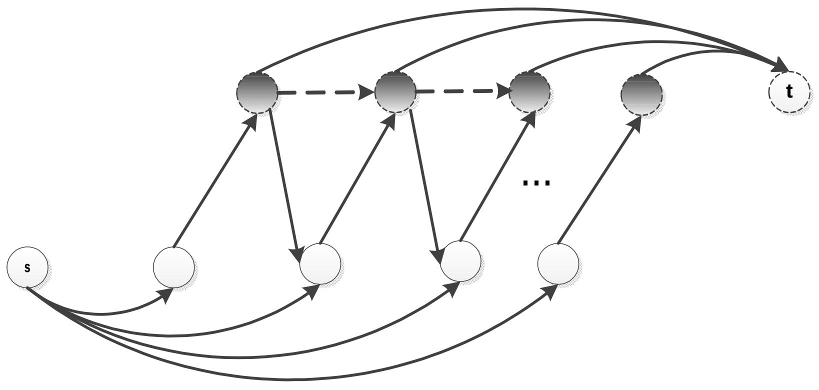


Figure 10 Modified network representation of crew scheduling instances in Beasley and Cao (1996)

In the tables that represent our results, the name of each instance consists of the following fields: number of nodes, number of paths to be balanced which is the result of the associated minimum capacity problem with the assumption that each duty requires one crew, policy in adding source and sink nodes and whether duties have costs or not. In the third field, “a” is used to show that source arcs emanate from the source node and end at all on-duty nodes, and sink arcs emanate from all tie-up nodes and end at the sink node. “s” is used to show that source arcs emanate from the source node and end at all on-duty nodes with no other incoming arcs, and sink arcs emanate from all tie-up node with no other outgoing arc and end at the sink node. In the fourth field, “c” is used to show that each duty has a cost equal to the length of the duty.

The results of the exact mathematical programming formulation and BSA for these instances are shown in Table 1. A time limit of 24 hours is imposed for the CPU time. Despite our expectation, BSA does not dominate the exact mathematical programming problem solution, though in some cases BSA ends with an optimal solution earlier than the exact mathematical programming problem. CPLEX fails to solve csp80-20-s-c instance with an “out of memory” error. In order to show the effect of using the improved interval for BSA, results for BSA on $[0, U]$ are shown. Except two instances - csp80-20-s, csp80-20-s-c- BSA with improved interval has better CPU time compared to BSA on $[0, U]$ interval.

Table 2 shows results of the conventional heuristic algorithms for crew scheduling instances. The heuristic algorithms are coded with two fields. In the first field “S” is used for static and “D” is used for dynamic methods. In the second field, “FL” stands for fixed-list, “CG” stands for comparative-gap and “SG” stands for sorted-gap methods. All algorithms end in less than a second with an optimal or near optimal solution. S-FL, D-FL, S-CG and D-CG algorithms using crisscross operator and D-FL and D-CG algorithms with hybrid operator find the optimal solution in all instances. As a result, the crisscross operator dominates the swap operator in all instances.

Table 1 Result of the exact formulation solution and BSA for crew scheduling instances in Beasley and Cao (1996)

			CPLEX				BSA						BSA [0, Z_{max}^{UB}]	
Prob. name	(#Var, #Cons)	Min Total WL	Obj.	Z_{min}	Z_{max}	Time	Obj.	Z_{min}^{LB}	Z_{min}^{UB}	Z_{max}^{LB}	Z_{max}^{UB}	Time	Obj.	Time
csp50-13-a	(4760,597)	7440	993	0	993	9 s	993	0	0	993	993	14 s	993	3.11 m
csp50-13-s	(3655,558)	10693	686	307	993	14 s	686	307	307	993	1190	11 s	686	1.68 m
csp50-13-a-c	(16216,1169)	13838	1499	161	1660	24 s	1499	161	161	1660	1705	1.11 m	1499	15.75 h
csp50-13-s-c	(5072,1091)	17304	948	712	1660	28 s	948	572	712	1660	1794	2.36 h	948	2.94 h
csp80-20-a	(21456,1581)	6706	536	0	536	3.16 m	536	0	0	536	647	3.56 m	536	14.28 m
csp80-20-s	(17301,1481)	8979	508	229	737	20.56 m	508	202	229	737	737	27.31 m	508	14.48 m
csp80-20-a-c	(25322,2761)	16739	1160	34	1194	42.6 m	1160	34	34	1194	1299	2.25 m	1160	10.69 h
csp80-20-s-c	(22642,2681)	20090	-	-	-	-	[485.492]	512	734	1222	1401	L:24 h	488	22.14 h
csp100-20-a	(29422,2241)	8841	658	0	658	6.93 m	658	0	0	658	804	2.01 h	658	5.44 h
csp100-20-s	(25422,2141)	13069	669	321	990	39.15 m	669	229	321	990	990	2.81 h	669	7.3 h
csp100-20-a-c	(34142,3721)	21344	-	-	-	L:24 h	1540	34	34	1574	1672	33.45 m	-	L:24 h
csp100-20-s-c	(30522,3641)	26650	566	1030	1596	5.38 h	-	707	1030	1596	1805	L:24 h	-	L:24 h

Table 2 Results of the conventional heuristic algorithms for crew scheduling instances in Beasley and Cao (1996)

	Crisscross						Swap						Hybrid		
Prob. Name	S-FL	D-FL	S-CG	D-CG	S-SG	D-SG	S-FL	D-FL	S-CG	D-CG	S-SG	D-SG	D-FL	D-CG	D-SG
csp50-13-a	993	993	993	993	993	993	993	993	993	993	993	993	993	993	993
csp50-13-s	686	686	686	686	686	686	698	698	698	698	698	698	686	686	686
csp50-13-a-c	1499	1499	1499	1499	1499	1499	1544	1544	1544	1544	1544	1544	1499	1499	1499
csp50-13-s-c	948	948	948	948	948	948	1145	1145	1145	1145	1145	1145	948	948	948
csp80-20-a	536	536	536	536	536	536	536	536	536	536	536	536	536	536	536
csp80-20-s	508	508	508	508	508	508	535	535	535	535	535	535	508	508	508
csp80-20-a-c	1168	1168	1168	1168	1168	1168	1225	1225	1225	1225	1225	1225	1168	1168	1168
csp80-20-s-c	488	488	488	488	488	488	797	797	797	797	797	797	488	488	488
csp100-20-a	685	685	685	685	700	700	720	720	720	720	720	720	685	685	700
csp100-20-s	669	669	669	669	669	669	761	761	761	761	761	761	669	669	669
csp100-20-a-c	1540	1540	1540	1540	1540	1540	1540	1540	1540	1540	1540	1540	1540	1540	1540
csp100-20-s-c	566	566	566	566	571	577	981	981	981	981	981	981	566	566	623

6.2. TCDD Instances

A set of instances from Turkish State Railways (TCDD) is used. Istanbul, Ankara and Eskisehir are three main regions of TCDD. In each of these regions, for planning period of one week instances of WBP are generated. Table 3 shows the result for the exact mathematical programming formulation solution and BSA. The second field in name of instances shows the number of paths to be balanced which is the result of the associated minimum capacity problem. A time limit of 24 hours is imposed for both algorithms. BSA is not able to find any results in 24 hours for these instances.

Table 3 Result of the exact formulation solution and BSA for TCDD instances

Prob. Name	(#Var, #Cons)	Min Total WL	CPU Time	CPLEX	BSA
Istanbul-38	(246052,23093)	1570.75 h	5.65 h	30.1 h	-
Ankara-50	(180052,23263)	2660.69 h	14.98 m	60.81 h	-
Eskisehr-65	(348337,37446)	3951.63 h	12.98 h	49.18 h	-

Table 4 shows the results of conventional heuristic methods for TCDD instances. All algorithms terminate with solutions which are better compared to the results of the exact mathematical programming problem. D-FL and D-CG algorithms using hybrid operator give the best results.

Table 4 Results of the conventional heuristic algorithms for TCDD instances

	Crisscross						Swap						Hybrid		
Prob. name	S-FL	D-FL	S-CG	D-CG	S-SG	D-SG	S-FL	D-FL	S-CG	D-CG	S-SG	D-SG	D-FL	D-CG	D-SG
Istanbul-38	8.08 h	8.08 h	8.08 h	8.08 h	14.93 h	13.63 h	12.05 h	12.05 h	12.05 h	12.05 h	18.56 h	18.56 h	6.95 h	6.95 h	13.1 h
Ankara-50	25.4 h	25.4 h	25.4 h	25.4 h	38.71 h	38.71 h	31.13 h	28.01 h	31.13 h	28.01 h	41.33 h	41.33 h	14.2 h	16.15 h	33.11 h
Eskisehr-65	14.31 h	13.43 h	12.51 h	13.43 h	36.01 h	31.31 h	33.75 h	33.75 h	33.75 h	33.75 h	35.76 h	40.3 h	9.25 h	7.3 h	29.65 h

Chapter 7

Conclusions and Future Research

The fairness issue is a critical planning phenomenon for all type of duty scheduling and rostering process where teams of crew members are in consideration such as railways and airlines. We study the fairness in terms of the workload of crew members. We define WBP as concerned with the assignment of crew members to as sequence of duties in a finite planning horizon in such a way that the allocation of the workload among them is acceptably fair and almost equal when possible. In order to formulate WBP, we focus on BPP (Cappanera and Scutellà (2011)) which determines k node-disjoint paths on a weighted network. Based on the suggested network representation in Şahin and Yüceoğlu (2011), we develop a network flow formulation of WBP ispired from the formulation of BPP. Due to the deadheading issue in trasportation crew scheduling, the formulation of WBP is indeed quite different from that of BBP.

We develop an exact algorithm to solve WBP based on a binary search method. Improvements on the search interval of BSA are discussed. The feasibility check procedure used in BSA is based on the cost expanded network of the original network. Using a property of the cost expanded network, a feasibility problem is developed for the feasibility check procedure. Some properties of the optimal solution of WBP are used to speed up BSA. Moreover, a pool of conventional heuristic methods based on a neighborhood search method is developed. Crisscross and swap operators as well as different selection criteria lead to multiple versions of such heuristic methods. We perform a computational study with well-known problem instances from crew scheduling literature and data sets that are representative of largest crew regions in TCDD. Our computational study shows that the proposed exact mathematical programming formulation of WBP (using CPLEX solver) is not capable of solving WBP on large networks to optimality within reasonable computational time. One the other hand, BSA does not dominate the exact mathematical programming formulation. For both sets of the problem instances, conventional heuristic methods are able to find either an optimal or a near optimal solution within a second. In conclusion, the computational study indicates that heuristic approaches are more efficient and effective for WBP.

For future research, we consider the fairness in terms of the allocation of different type of duties among crew as a valuable venue. In addition, minimizing operational costs of unbalanced allocation of different type of duties is an open research question to be studied. From the methodological point of view, different solution approaches for WBP such as a column generation approach etc. can be investigated.

Bibliography

[1] Abbink, E., Wout, J.V., Huisman, D., Solving Large Scale Crew Scheduling Problems by Using Iterative Partitioning, In: Liebchen, C., Ahuja, R.K., Mesa, J.A. (Eds.), *ATMOS 2007*, (2007).

[2] Beasley, J. E. and Cao, B., A Tree Search Algorithm for the Crew Scheduling Problem, *European Journal of Operational Research*, 94(3), 517–526 (1996).

[3] Bellanti, F., Carello, G., Della Croce, F., Tadei, R., A Greedy-Based Neighborhood Search Approach to A Nurse Rostering Problem, *European Journal of Operational Research,*153, 28-40 (2004).

[4] Burke, E., Cowling, P., Causmaecker, P. D., Berghe, G. V., A Memetic Approach to the Nurse Rostering Problem, *Applied Intelligence*, 15, 199–214 (2001).

[5] Cappanera, P. and Scutellà, M., Balanced Paths in Acyclic Networks: Tractable Cases and Related Approaches, *Networks*, 45(2), 104-111 (2005).

[6] Cappanera, P. and Scutellà, M., Color-Coding Algorithms to the Balanced Path Problem: Computational Issues, *INFORMS Journal on Computing*, 23, 446-459 (2011).

[7] Chiarandini, M., Schaerf, A., Tiozzo, F., Solving Employee Timetabling Problems with Flexible Workload using Tabu Search, In: Burke, E. and Erben, W. (Eds.), *Proceedings of PATAT 2000*, (2000).

[8] Suyabatmaz, A. Ç. and Şahin, G., A Column-and-Row Generation Algorithm for A Crew Planning Problem in Railways, In: Klatte, D., Lüti, H. J., Schmedders, K. (Eds.), *Proceedings of OR 2011*, 335–340 (2011).

[9] Şahin, G. and Yüceoğlu, B., Tactical Crew Planning in Railways, *Transportation Research Part E: Logistics and Transportation Review,* 47, 1221–1243 (2011).