Proceedings of the International MultiConference of Engineers and Computer Scientists 2012 Vol II,
IMECS 2012, March 14 - 16, 2012, Hong Kong

# Slacken Piercing Point Based Subproblem Algorithm for the LP Relaxation of Crew Rostering

Zhao Mingyu, Liu Qiong and Wang Zhenyu

*Abstract*—**Crew rostering is one of the most important scheduling problems that airlines have to face regularly. The goal is to assign a set of tasks to a group of workers as fairly as possible while following various business rules. Difficulties of the optimization mainly arise from its large scale. In this paper, a modified subproblem based algorithm for the LP relaxations of crew rostering problem is proposed, whose benefit comes from two main ideas. Firstly, it needn't consider all non-basic variables in any iteration. Secondly, it is beneficial to consider primal and dual problems at the same time. The computational experiments on the real-world problems showed that time savings about 50% over previous universal subproblem based algorithm can be obtained.**

*Index Terms*—**crew rostering optimization, set partition problem, subproblem algorithm, probe operation**

## I. Introduction

CREW rostering is an evergreen research topic for large airlines, which is to assign a set of tasks to a group of workers as fairly as possible while following various business rules. The readers are referred to [1] for an outline about the airline operations from the operation research point of view, and to [2], [3] for details on the crew rostering problem (CRP). The CRP is typically formulated as an extension of the set partitioning problem (SPP), and the branch-and-bound based heuristic is usually used to give integer solution. It makes the LP relaxation one of the most import building blocks in the branch-and-bound, and led to the subject of this study.

Difficulties of CRP mainly arise from its huge scale, even though for its LP relaxations. The magnitude of variables can be up to several hundred million in one model for a large airline's monthly problem. To overcome the difficulty, two main methods have been proposed. The first is the column generation method, such as [4]. It requires some properties that can't be possessed by the real-world problems. For example, it assumes that the cost of a roster must be decomposed into tasks or combinations of tasks in sequence. As the business rules are complex and mutable, it isn't satisfying usually, so the subproblem can't be modeled accurately as a shortest path problem with resource constraints. The second divides scheduling time zone into multiple short stages, whose size is controlled by the so-called generation-and-optimization strategy, and circularly improves the subproblems in each stage until reaching a satisfying solution for the global problem. The comparisons of these two methods can be found in [3]. For a large airline, the latter is more practical. Nevertheless, even for the subproblem, the magnitude of variables can still be up to $10^6$ or $10^7$. Although the modern LP software has received prominent improvement in the past decade, it still has to consume a few hours to solve an instance with several millions of variables.

Maybe the most successful story for the large scale SPP was the subproblem based approaches [5], [6], which saved the time and space requirements prominently. This term subproblem has different meaning from that of the column generation. It has the same structure as the original, but only consists of considerably less variables, and its solution is also feasible for the original. In some sense, the subproblem approach can be view as an extension of the partial price. In each steps, it solves the current subproblem with simplex. And then all non-basic variables are divided into two subsets based on some local information. The "good" columns are regarded as the candidates bringing the furthest convergence from a global view, as opposed to the "bad" columns which needn't be considered in the next. It follows that the "good" columns are added to the next subproblem. Repeat the process until attaining the optimal of the original problem. The foremost distinguish among different subproblem algorithms is how to identify the "good" columns from all non-basic variables at each iteration, such as the SPRINT [5] based on the classical Dantzig rule, whereas the PAPA [7] and the P-D subproblem algorithm [6] based on the piercing point concept.

In this study, a modified column selection criterion for the LP relaxations of CRP was proposed, based on the special structure of CRP. By the modified algorithm tuned to the CRP, it can save about half of the solving time with compare to the previous subproblem algorithms.

This paper is organized as follows. Section 2 formulates the CRP and defines a number of symbols used throughout this paper. Section 3 describes the main column selection criterion for CRP and gives the formulation of the algorithm , which is followed by the experiments and analysis in Section 4. Some conclusions are drawn in section 5.

## II. Crew rostering problem

Let $C$ be the set of crew members, and $T$ be the set of tasks to be assigned. A legal roster is a set of tasks for some crew member, following legislations and various business rules, and is referred to as LoW (Line of Work), denoted as $\mathbf{a}_j$. Each LoW must belong to a crew member $f(\mathbf{a}_j) \in C$.

Zhao Mingyu is with the School of Software Engineering, South China University of Technology, Guangzhou, Guangdong 510640, P.R.China, email: zhmy@21cn.com
Liu Qiong is with South China University of Technology. email:liuqiong@scut.edu.cn
Wang Zhenyu is with South China University of Technology. email:wangzy@scut.edu.cn

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

**Algorithm 1:** The modified algorithm for Crew Rostering Problem

$A' \leftarrow \{\mathbf{a}_j \,|t(\mathbf{a}_j) \leqslant 1, f(\mathbf{a}_j) \in C\} \cup \{\mathbf{a}_j \,|f(\mathbf{a}_j) \notin C\}, \pi \leftarrow \mathbf{0}$
**repeat**
  Solve the subproblem $\min z = \mathbf{cx}$ s.t. $A'\mathbf{x} = \mathbf{b}$, let $\rho$ denote its optimal dual vector, and $S \subseteq C$ consists of crews that have changed in the last basis.
  $\tau \leftarrow \max \left\{ 0, \frac{-d_j^{\rho}}{d_j^{\pi} - d_j^{\rho}} \middle| d_j^{\rho} < 0, d_j^{\pi} > 0, f(j) \notin S \right\}$
  $\pi \leftarrow \tau \cdot \pi + (1 - \tau) \cdot \rho$
  Remove all non-basic columns from $A'$
  **if** $z > \pi\mathbf{b}$ **then**
    $L \leftarrow \emptyset$
    **foreach** $\mathbf{a}_j \notin A' \wedge d_j^{\pi} \geqslant 0$ **do**
      **if** $d_j^{\pi} = 0$ **then**
        $A' \leftarrow A' \cup \{\mathbf{a}_j\}$
      **else if** $f(\mathbf{a}_j) \notin S$ *and* $(d_j^{\pi} \leqslant \epsilon^{+} \wedge d_j^{\rho} \geqslant 0) \vee (d_j^{\pi} \leqslant \epsilon^{-} \wedge d_j^{\rho} < 0)$ **then**
        $L \leftarrow L \cup \{\mathbf{a}_j\}$
      **end if**
    **end foreach**
    Add the most $\delta N_s$ columns to $A'$ which possess small reduce cost with $\pi$ in $L$
    Add the most $(1 - \delta)N_s$ columns to $A'$ which possess small reduce cost with $\pi$ and then $\rho$ in $L$
  **end if**
**until** $\tau = 0$;
SPRINT()

Especially, the empty LoW (not include any task) is legal for any crew member. For any LoW $\mathbf{a}_j$, it consists of a number of tasks, and its cardinality is denoted as $t(\mathbf{a}_j)$. Each LoW $\mathbf{a}_j$ has an associate cost $c_j$ evaluating its fairness.

Crew rostering problem is typically formulated as an extension of the set partitioning problem, in which we want to find a minimum cost subset of the LoWs following three categories of constraints. The first is that each crew member has to choose one and only one LoW from all LoWs belonging to her/him, which is referred to as assignment constraint. Second, the crew members assigned to any task should not exceed its capacity. Third, some tasks must have crew members more/less than some amount, whose qualification should be as desire. Every LoW can form a column according to these constraints, and properly introduces some supplier and slack variables additionally, with which the constraint matrix $A$ can be constructed. As almost every column in $A$ indicates a LoW $\mathbf{a}_j$, it is also denoted as $\mathbf{a}_j$ for simple representation. Decision variable $x_j$ is equal to 1 if LoW $\mathbf{a}_j$ is included in the solution, and 0 otherwise.

In summary, the relaxation of crew rostering problem can be formulated as a LP, see (1). The readers are referred to [3] for details on the real-world business rules and the corresponding relations with the model.

$$(\mathrm{P}) \quad \begin{array}{ll} \min & \mathbf{cx} \\ \text{s.t.} & A\mathbf{x} \leqslant \mathbf{b} \\ & \mathbf{x} \geqslant \mathbf{0} \end{array} \tag{1}$$

Without loss of generality, it can be assumed $\mathbf{c} \geqslant \mathbf{0}$ and $\mathbf{b} \geqslant \mathbf{0}$. A subset of all LoWs and some other variables constitute a subproblem of (P), whose solution is feasible to (P) and denoted as (SP).

## III. The modified algorithm for CRP

In this section, a modified subproblem algorithm for CRP based on a slacken piercing point concept was proposed, with an efficient subproblem construction strategy adapted to this special problem. The formal description of the modified algorithm was given by Algorithm 1.

### A. Initial solution

In the real world environment, the rostering problem hasn't any initial feasible solution usually. The first subproblem consists of not only all single task LoWs and empty LoWs for all crew, but also all supplier and slack variables. Evidently, it can produce a feasible solution for the original problem, because it is legal to assign empty LoW to all crew members at least. In addition, the vector $\mathbf{0}$ is a dual feasible solution as the cost of any column is non-negative.

### B. Slacken piercing point

As mentioned before, the main difference between subproblem algorithms is how to construct the next subproblem at each iteration. Like price scheme for the simplex, different subproblem construction strategies have dramatically impact on the overall performance for the subproblem based method, as indicated by [5], [6]. As a result, the subproblem construction method is certainly the kernel subject for a subproblem based algorithm.

Let $\pi$ denote the incumbent dual vector, and $\rho$ the optimal dual solution of the current subproblem. The piercing point $\pi'$ can be written as $\pi' = \theta\pi + (1 - \theta)\rho$, where $\theta$ makes $\pi'$ remain dual feasible and improve the dual object $\pi'\mathbf{b}$ as large as possible. A probe is the operation of finding the piercing points and the columns that decide $\theta$. Those columns could be regarded as the best candidates for the next subproblem. These concepts were defined by [7] with dual version.

It is well known that it is possible to design more efficient price algorithms for a special class of problems which exhibit some identifiable structures. For the rostering problem, every crew member must have one and only one LoW in any feasible solution, as a consequence of the assignment constraints. Thus, every crew member has several LoWs in any basic feasible solution for its LP relaxation. As for some crew member $k \in S \subseteq C$, if her/his LoWs in current basis have changed since the last iteration, we attempt to keep these LoWs in the next basis. The idea behind is that it has


ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

got locally "best" representations for crew $k$, so it is unlikely that $k$ is still a good improving candidate in the next iteration, i.e. the best candidate columns are unlikely to come from this crew member. So, this crew member won't be considered in the probe operation any more. To distinguish with $\theta$, $\tau$ is defined as bellow

$$\tau = \max\left\{0, \frac{-d_j^\rho}{d_j^\pi - d_j^\rho} \middle| d_j^\rho < 0, d_j^\pi > 0, f(j) \notin S\right\} \quad (2)$$

where $d_j^\rho$ and $d_j^\pi$ denote reduce costs of column $\mathbf{a}_j$ with $\rho$ and $\pi$ respectively. The updated dual vector $\pi' = \tau\pi + (1-\tau)\rho$ is referred to as the slacken piercing point of $\pi$ and $\rho$. In each iteration, the columns are regarded as the best candidates which have zero reduced cost with slacken piercing point. These columns limit the further improvement of current dual object, so they are referred to as restrict variables for simplify representation.

### C. Column selection

By the definition of slacked piercing point, it is evident that all restrict variables should be added to the subproblem. Nevertheless, there are only a few of restrict variables in each iteration usually. It nearly can't work if only these restrict variables are added to subproblem for a large problem. Thus, we have to choose some other potentially beneficial columns.

In fact, as indicated by [5], [6], the size of subproblem has an important impact on subproblem based methods, as the more columns are introduced to the next subproblem, the more solving time will be needed inevitably. More importantly, a too greedy strategy based on local information will lead to unnecessary simplex iterations from a global view. In contrast, too few columns will result in too many redundant pricing. So, the subproblem size should be traded off for this modified algorithm carefully.

Although the goal of the slacken piercing point based criteria is to satisfy the complementary slackness in a set which is made of columns with approximately equal reduce costs with $\pi'$, the columns with smaller reduced cost with $\rho'$ are better. So, reduced cost with $\pi'$ and $\rho'$ is used to evaluate other columns simultaneously, rather than the P-D subproblem algorithm did.

In keeping with probe operation, other columns of the crew members that belong to $S$ won't be added to subproblem unless they are restrict variables. As for a member not belonging to $S$, the candidate set $L$ consists of columns whose reduced cost with $\rho$ is less than zero and reduced cost with $\pi$ is less than or equal to $\epsilon^-$, and the columns whose reduced cost with $\rho$ is positive and reduced cost with $\pi$ is less than or equal to $\epsilon^+$, where $\epsilon^+ < \epsilon^-$. That is,

$$L = \left\{\mathbf{a}_j \middle| d_j^\rho \geqslant 0, d_j^{\pi'} \leqslant \epsilon^+ \text{ and } f(j) \notin S\right\} \cup \left\{\mathbf{a}_j \middle| d_j^\rho < 0, d_j^{\pi'} \leqslant \epsilon^- \text{ and } f(j) \notin S\right\} \quad (3)$$

Several parameters are used to control the subproblem size in the modified algorithm. To construct the next subproblem, it is firstly selected no more than $\delta N_s (0 \leqslant \delta \leqslant 1)$ columns with smallest reduced cost with $\pi'$ from $L$, regardless of $d_j^\rho$. And then it is added at most $(1-\delta)N_s$ columns with smallest reduced cost with $\pi'$ and $d_j^\rho < 0$ from $L$. That is, some columns with $d_j^\rho < 0$ are used instead of some with smaller reduced cost with $\pi'$.

To efficiently select columns from $L$, two dimensional buckets are used to rank columns, which have approximately equal reduce costs with $\pi'$ and $\rho$ placed in the same bucket. If $\mathbf{a}_h \in L_{i,j}$ and $\mathbf{a}_g \in L_{i+1,j}$ then $d_h^\pi \approx d_g^\pi$ and $d_h^\rho < d_g^\rho$, and if $\mathbf{a}_g \in L_{i,j+1}$ then $d_h^\pi < d_g^\pi$ and $d_h^\rho \approx d_g^\rho$.

### D. Optimality

Note that the slacken piercing point doesn't consider all columns in the probe operation, and the corresponding $\pi'$ may not be dual feasible for primal problem (P). Nevertheless, if the reduced cost of a column with the slacked piercing point is negative, it is still possible to become non-negative in the later iterations. Although it is possible that we can't obtain the optimum of the original problem by this method after satisfying condition $\tau = 0$, it is usual that the incumbent is very close to it.

If a column not dual feasible for $\pi$ exists after loop, call the SPRINT algorithm to guarantee the convergence. As for the SPRINT algorithm, it uses an array of appropriate size to accommodate the potential columns. The columns with negative reduced cost with $\rho$ are appended from the beginning of the array, and others less than some threshold (such as $10^{-5}$) are appended from the bottom to back. The columns with negative reduced cost can override the others, and not true in turn. The simplification is rooted in that the initial basis is very close to the optimal point.

### E. Analysis

There are several reasons why this approach is computationally efficient. First, it uses $\tau \leqslant \theta$ to update $\pi'$ in the probe operation so that a larger dual object value is obtained, although it is slightly not dual feasible for the original problem (P). At the same time, it adds more good columns for the current subproblem ($d_j^\rho < 0$), which will decrease the object value of the next subproblem more rapidly. Therefore, it converges to some good feasible point for (P) more quickly. Second, this strategy lightens the burden on the probe and sort operation in each iteration, especially beneficial to large scale problem. Finally, even a column may become not dual feasible in some iteration, it is not permanent. It still has the possibility of becoming dual feasible later. The convergence point of the P-D phase is usually so close to the optimum of the original that it can rapidly derive optimality by SPRINT algorithm.

## IV. Computational Experience

In this study, a number of problems were experimented on a 2.66GHz PC with 4GB of RAM, and Xpress [8] was used as the LP solver for the subproblems. These problems came from the operation control system of a large international airline in China, and targeted on different crew ranks. In fact, each rank has its model characteristic since the corresponding business rules are appropriative. Table I presents the numbers of crews and tasks with the scale of constraint matrix in the problems solved.

Table II presents each operation's time of the modified algorithm for every problem listed in Table I. Every problem

TABLE II
OPERATION TIME IN THE MODIFIED ALGORITHM FOR THE PROBLEMS

| Prob. | Major iter. | Simplex iter. | Simplex time | Probe | Price update | Price sort | SPRINT Iter. | SPRINT iter. | SPRINT time | SPRINT price | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| No.1 | 66 | 151796 | 15.08 | 2.73 | 3.54 | 4.23 | 3 | 7392 | 1.23 | 0.20 | 27.13 |
| No.2 | 129 | 245002 | 22.35 | 10.47 | 11.49 | 11.17 | | | | | 55.70 |
| No.3 | 77 | 260290 | 22.45 | 11.10 | 15.53 | 4.97 | 3 | 712 | 0.45 | 0.86 | 55.63 |
| No.4 | 54 | 394199 | 38.95 | 10.94 | 15.87 | 7.24 | 11 | 26634 | 6.84 | 2.50 | 82.80 |
| No.5 | 49 | 98661 | 8.17 | 13.84 | 17.58 | 11.01 | | | | | 51.42 |
| No.6 | 58 | 138609 | 11.33 | 23.02 | 27.65 | 13.75 | 2 | 24 | 0.17 | 1.20 | 77.83 |

TABLE IIIa
COMPARISON WITH P-D SUBPROBLEM SIMPLEX ALGORITHM FOR NO.1 $\sim$ 3

| | No.1 | | | No.2 | | | No.3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Major iterations | 379 | 66 | 82.6% | 429 | 129 | 69.9% | 90 | 77 | 14.4% |
| Solving (SP) time | 22.56 | 15.08 | 33.2% | 24.17 | 22.35 | 7.5% | 33.23 | 22.45 | 32.4% |
| Probe time | 22.90 | 2.73 | 88.1% | 43.13 | 10.47 | 75.7% | 20.22 | 11.10 | 45.1% |
| Update time | 20.58 | 3.54 | 82.8% | 38.15 | 11.49 | 69.9% | 18.29 | 15.53 | 15.1% |
| Sort time | 24.49 | 4.23 | 82.7% | 36.99 | 11.17 | 68.8% | 7.90 | 4.97 | 37.1% |
| Probe+Sort/iter. | 0.13 | 0.11 | 15.4% | 0.19 | 0.17 | 10.5% | 0.31 | 0.21 | 32.3% |
| SPRINT phase | | 1.43 | | | | | | 1.31 | |
| Total | 90.77 | 27.13 | 70.1% | 142.63 | 55.70 | 60.9% | 79.88 | 55.63 | 30.4% |

TABLE IIIb
COMPARISON WITH P-D SUBPROBLEM SIMPLEX ALGORITHM FOR NO.4 $\sim$ 6

| | No.4 | | | No.5 | | | No.6 | | |
|---|---|---|---|---|---|---|---|---|---|
| Major iterations | 129 | 54 | 58.1% | 107 | 49 | 54.2% | 149 | 58 | 61.1% |
| Solving (SP) time | 57.10 | 38.95 | 31.8% | 14.89 | 8.17 | 45.1% | 19.43 | 11.33 | 41.7% |
| Probe time | 40.87 | 10.94 | 73.2% | 41.47 | 13.84 | 66.6% | 76.11 | 23.02 | 69.8% |
| Update time | 38.14 | 15.87 | 58.4% | 38.91 | 17.58 | 54.8% | 71.59 | 27.65 | 61.3% |
| Sort time | 23.40 | 7.24 | 69.1% | 36.03 | 11.01 | 69.4% | 49.55 | 13.75 | 72.3% |
| Probe+Sort/iter. | 0.50 | 0.34 | 32.0% | 0.72 | 0.51 | 29.2% | 0.84 | 0.63 | 25.0% |
| SPRINT phase | | 9.38 | | | | | | 1.39 | |
| Total | 159.95 | 82.80 | 48.2% | 131.83 | 51.42 | 61.0% | 217.36 | 77.83 | 64.2% |

TABLE I
PROBLEMS FOR CREW ROSTERING

| Prob. | Crews | Tasks | Rows | Columns |
|---|---|---|---|---|
| No.1 | 372 | 208 | 1412 | 1,922,626 |
| No.2 | 263 | 208 | 1319 | 2,343,817 |
| No.3 | 356 | 274 | 1855 | 4,386,271 |
| No.4 | 584 | 323 | 1859 | 9,412,213 |
| No.5 | 476 | 237 | 1224 | 12,446,241 |
| No.6 | 485 | 239 | 1237 | 15,761,156 |

was solved through setting $N_s = 2 \times 10^4$, $\epsilon^+ = 200, \epsilon^- = 400$ and $\delta = 0.8$. These results showed that the optimal solution was usually very close to that of the primal problem after solving the relaxed problem. It also showed that the algorithm worked well for various sizes of problems and for various models that targeted on different crew ranks.

Table IIIa,IIIb give comparisons of each operation with the P-D subproblem simplex algorithms. For each problem, the first column indicates the P-D subproblem algorithm, and the second indicates the modified algorithm. The third indicates improved percentages yielded by the modified algorithm which equals to the difference of column 1 and 2 divided by the first column. For the P-D subproblem algorithm, the parameter $N_s$ was also set to $2\times10^4$, while any column with $d_j^\pi > 300$ couldn't join the subproblems.

There are some results shown in Table IIIa,IIIb. Firstly, the improvement of the modified algorithm mainly came from its dramatically decrease of the iterations. This proves that the columns selection criteria of the modified algorithm is more effective and efficient. Secondly, the improvement of the probe and sort operation (interpreted as "Probe+Sort/iter.") also contributed largely to the overall performance, especially for the large scale problems.

## V. CONCLUSION

In this study, we have modified the concept of probe operation and proposed a modified column selection criteria for the crew rostering problem taking place in airlines, which can be used to design a subproblem based algorithm. The idea of the modification is to divide variables into groups, and only price columns in the most promising group at each step. At the same time, consider two factors, one for primal problem and the other for its dual, to select columns.

The computational experiments on the real-world problems have been presented, which have shown that the modified algorithm saved about one-half time with compare to the primal-dual subproblem algorithm.

## Acknowledgment

The authors would like to thank some large airline in China for providing us with the real-world problems.

We are also appreciative of the valuable feedback and comments from the anonymous referees.

## References

[1] Cynthia Barnhart. "Airline Scheduling: Accomplishments, Opportunities and Challenges." Lecture Notes in Computer Science, Vol. 5015, 2008.

[2] Cynthia Barnhart, Amy Cohn, Ellis Johnson, Diego Klabjan, et.al. "Airline Crew Scheduling." Randolph W. Hall, ed. Handbook of Transportation Science, Kluwer Scientific Publishers, Boston. 2003, pp. 57-93.

[3] Niklas Kohl, Stefane E. Karisch. "Airline Crew Rostering: Problem Types, Modeling and Optimization." Annals of Operations Research, Vol. 127, No. 1-4, 2004, pp. 223-257.

[4] Gamache, M., F. Soumis. "A method for optimally solving the rostering problem." G. Yu, ed. Operations Research in the Airline Industry. Kluwer, Boston, MA, pp. 124-157

[5] Ranga Anbil, Rajan Tanga, Ellis Johnson. "A Global Approach to Crew Pairing Optimization." IBM Systems Journal 31,1991, 71-78.

[6] Jing Hu, Ellis Johnson. "Computational results with a primal-dual subproblem simplex method." Operations Research Letters, Vol. 25, No. 4, 1999, pp. 149-157.

[7] Awanti P. Sethi, Gerald L. Thompson. "The pivot and probe algorithm for solving a linear program." Mathematical Programming, Vol. 29, No. 2, 1984, pp. 219-233.

[8] FICO. "Xpress-Optimizer Reference manual." Release 22.01. 2011.

ISBN: 978-988-19251-9-0
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)