グラフの色分け手法の車両運用問題への適用

外山春彦、荒木 大
株式会社 東芝 研究開発センター

予め作成された計画に事故などにより変更が生じる場合がある。このとき、与えられた制約条件を満足するように、かつ、出来るだけ少ない変更で、計画のほかの部分を修正する必要が生じる。我々はこの問題を「リソース再割り当て問題」として定式化し、グラフの色分け手法を適用する方法を考案した。計画修正範囲を出来るだけ少なくすることを、修正範囲を徐々に広げながら探索を行う機構と元の計画を初期解とする近傍探索機構を組み合わせる事で実現した。この方法を「鉄道車両運用計画の修正」に適用し、出来るだけ少ない修正で元の計画に戻すという観点で、人手よりも良い修正計画を短時間に作成できる事を確認した。

Train Schedule Revision by Graph Coloring Method

Haruhiko Toyama, Dai Araki
TOSHIBA Corporation, Research and Development Center

We propose a method for schedule revision on resource assignment problem. Some accidents causes some changes in the original schedule. Then, we have to revise the schedule with less changes to satisfy the given constraints. We formulate this problem into Resource Reassignment Problem. For this problem we propose a method using Graph Coloring Method. To revise the schedule with less changes, this system has two mechanisms. The one is a search mechanism widening the schedule revision range gradually, out of which the revised schedule is same as the original one. The other one is a local search mechanism with the original schedule initial data. We apply this method to Train Schedule Revision Problem, and we can get schedules revised with less changes, than that revised manually.

1. はじめに

与えられた作業（ジョブ）に対して、様々な制約条件を考慮しながら資源（リソース）を割り当てる問題は「リソース割り当て問題」と呼ばれ、その応用例としては、鉄道・バス・運送会社における乗務員運用計画や車両運用計画、工場の設備運用、あるいは時間割作成などがある。本稿では「車両運用計画の修正」という問題を扱う。これは、予定されていた車両運用計画に事故や運転支障によって変更が生じた場合に、変更発生日以降の運用計画を見直す問題である。このような問題は一般的に「リソースの再割り当て問題」として定式化できる。つまり、リソースの割り当て計画に対して部分的な変更が加わったときに、与えられた制約条件を満足するように、かつ、できるだけ少ない変更で、計画の他の部分を修正する問題である。

本稿では、リソースの再割り当て問題にグラフカラーリングを適用する方法を提案し、それを車両運用計画問題へ応用した事例を紹介する。基本的な考え方は、計画変更を容認するジョブの範囲を徐々に広げながら、できるだけ少ない範囲で計画修正が留まるように探索を行う。個々の探索範囲で計画変更の影響が吸収できるか否かの判定にグラフカラーリングを用いる。すなわち、探索範囲内のジョブ間の制約条件をグラフで表現し、そのグラフを色分けすることでリソースの割り付けを決める。さらに、グラフカラーリングには近

傍探索手法の一種である Life Span Method(LSM)[3]を改良して用い、その初期解に元の計画を用いることで、個々の探索範囲内での計画修正をさらに少なくすることが可能になる。このように計画変更容認する範囲を徐々に広げる仕組みと、その範囲内での計画修正をできる限り少ない修正で行う仕組みとをうまく組み合わせることにより、短時間で、できるだけ少ない計画修正を行うことが可能なシステムを組むことができる。試作した車両運用計画修正システムにおいて、出来るだけ少ない修正で元の計画に戻すという観点で、人手よりも良い修正計画を短時間で作成できる事を確認した。

2. リソース再割り当て問題へのグラフ色分け手法の応用

2.1. リソース割り当て問題へのグラフカラーリング手法の適用

列車の車両運用はリソースの割り当て問題の一種である。ジョブは列車の運行であり、出発駅・開始時刻と終着駅・到着時刻の組みでデータが与えられる。これらの運行に対してどの列車編成(リソース)を割り当てるかを決める事が車両運用計画である。リソース割り当ての典型的な制約条件は、「作業時間が重なっているジョブに同じリソースを割り当てる事はできない」である。これ以外に、車両運用計画に特有な制約条件としては「前のジョブ(運行)が終わる場所と次のジョブ(運行)の始まる場所が異なるような 2 つのジョブに同じリソース(列車編成)を割り当ててはいけない」「一定期間ごとにリソースは検査を行う。検査中はジョブにそのリソースを割り当てる事は出来ない」といった条件がある。

このようなリソース割り当てに、グラフカラーリングを用いる事が出来る。グラフカラーリングとは、ノードとそれらを結ぶ辺からなるグラフ上で、隣接するノードが必ず別の色になるように各ノードに色を割り振る手法である。リソース割り当て問題をグラフカラーリングで解くには、まず、与えられたジョブをグラフのノードとみなし、同一リソースの割り当てが出来ないジョブのペアを辺で結ぶ。次に、このグラフをリソースの数の色でカラーリングを行う。カラーリングで得られた結果を元に、同一色のノードを同一のリソースへ割り付けることで、リソースの割り当てを得る事が出来る。

2.2. リソースの再割り当て問題

本稿では、リソースの割り当てが部分的に変更されたときに、与えられた制約条件を満足するように、かつ、出来るだけ少ない変更で、計画のほかの部分を修正する「リソース再割り当て問題」を考える。修正の少なさの評価には「出来るだけ短い期間でもとの計画に戻る」「元々計画変更のなかったリソースに対しての修正の影響を少なくする」といった評価基準が考えられる。

このように、出来るだけ小さい計画修正により不測事体に対する対応を行う方法は状況対応型のスケジューリングと呼ばれている[1],[2]。状況対応型のスケジューリングの解法として制約指向のアプローチから、ペナルティ伝播法[1],[2]等の手法が提案されている。

本稿では、これらの既存アプローチとは異なり、リソース再割り当てをグラフカラーリングの手法を用いて行う方法を提案する。

2.3. グラフカラーリングによるリソース再割り当て手法

グラフカラーリングを用いてリソースの再割り当てを行う基本アルゴリズムを以下で説明する。

ステップ 1: 計画修正の探索範囲を設定する。

計画を修正する探索範囲を設定し，この範囲外は元の計画どおりであるとする。この探索範囲には，初期

事象，つまりリソース割り付けが変更されたジョブは必ず含める。探索範囲は，始めは狭く設定する(後で説明する車両運用修正問題では，運用変更が発生した日と翌日のジョブを第 1 回目の探索範囲とした)。以下のステップでその探索範囲内だけで計画修正の収束が可能であるかを調べる。これに失敗した場合は，成功するまで徐々に探索範囲を広げる(車両運用修正問題では半日づつ範囲を広げた)。

ステップ 2: 探索範囲内のジョブをグラフで表現する。

探索範囲内のジョブをグラフのノードで表す。次に，同一のリソースを割り付けることができないジョブに対応するノードの間を辺で結ぶ。初期事象(リソース割り当てが変更されたジョブ)と、探索範囲の外のジョブと接続関係を持つ境界領域のジョブについては、元の計画通りのリソースを割り当てるという仮定をおき、同じリソースを割り付けるジョブは一つのノードにまとめて表す。また、その外の制約条件から同一のリソース割り当てが必要なジョブのペアも一つのノードで表す。

ステップ 3: グラフに色を割り付ける。

リソース数だけの色を用いて，隣接するノードが必ず異なる色になるように色分けを行う。成功した場合はステップ 4。失敗したら，ステップ 1 に戻って探索範囲を広げて再実行する。

ステップ 4: 色とリソースの対応を決め，ジョブへのリソースの割り付けを決定する。

ジョブに割り付けた色とリソースとを対応させることによってリソースの再割り付け結果が得られる。

探索範囲の設定に「計画の変更を許容する期間(計画を元に戻す目標日)」を今回は用いた。この場合、「計画の変更がなかったリソースへの影響を少なく」する工夫が必要である。そこで、ステップ 3 で用いるグラフカラーリング手法に局所探索手法である Life Span Method を用い、元の計画を初期値とすることで対処した。

2.4. Life Span Method

グラフカラーリングアルゴリズムには Life Span Method [3]を修正して用いた。このアルゴリズムは、ノード数が n のグラフを k 色で塗り分ける手法である。山登り的な近傍探索手法であるため、元の計画での割り当てから決まる色の割り付けを初期値として探索を行うことで、修正範囲の少ない計画修正を行うことが可能になる。アルゴリズムを述べるために以下の記号を用いる。

V={V1,...Vn} ={グラフのノード全体}

color(i) := ノード Vi に割り付けられた色。k 色の色は整数 1 ,..., k で表す。

C(i, m) := ノード Vi に隣接するノードで，色 m が割り付けられているノードの個数。

bad-deggree(i) := C(i, color(i)) = ノード Vi に隣接する同じ色のノードの個数

パラメータとして，min-tabu-length, max-tabu-length、α の 3 つを与える。

以下に[3]で提案されている、k 色に色分けする手続きを示す。

```
1:  procedure Fixed-k LSM
2:      各 i = 1, ..., n に対して, color(i) へ 1,..,kの整数を適当に割り付ける
3:      各 i = 1, ..., n、m = 1,...,k に対して,  LS(i) := 0, LTM(i,m) := 0
4:      while 色分けが完了してない do
5:          Φ := { ( i, m )| i = 1, ..., n; m = 1,..., k; bad-degree(i) >0 かつ LS(i) = 0 }
6:          F(i, m):= C( i, m ) - bad-degree( i ) + α  ×LTM(i, m)
7:          F(i, m)を最小にする(i, m) ∈ Φ を選び(i*, m*)とする (複数ある場合は乱数で選択)
8:          color( i*) := m*
9:          LS(i*) := min-tabu-length から max-tabu-length までの乱数
```

```
10:          各 i ∈{1,... ,n; LS(i)>0}に対して,   LS(i) := LS(i) - 1
11:          LTM(i*, m*) := LTM(i*, m*) +1
10:  return
```

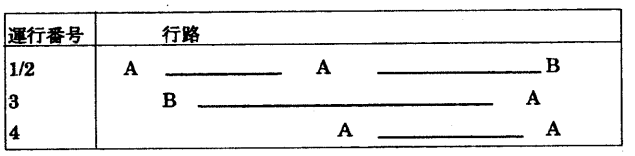
3. 車両運用計画修正システムの実現

3.1. 車両運用計画修正問題について

車両運用計画とは、どの列車編成がどの運行を担当するかを決めた計画であるが、予定していた車両運用計画に事故や運転支障による変更が生じたときに、支障発生日以降の運用計画を見直す作業(車両運用計画の修正)が発生する。通常、支障発生日以降の計画を完全に作り直すのではなく，できるだけ早い日数で元の運用計画に戻るように計画修正を行う。これは，車両検査の都合が主な理由であり，走行距離数に応じた検査計画の予定を車両ごとにあらかじめ長期間にわたって作っており，これを変更することが難しいからである。車両を交換できる場所が限られている事もあり、車両運用の変更が発生すると、簡単には元の計画に戻すことはできない。実際に事故等が発生した場合には，翌日以降の数日から数週間にわたって当初の計画とは異なる計画を用いなければならない。

車両運用計画の例

一日分の列車の運行は次の様な列車運行表で表される。



ここで，横方向に一日の時刻を表す。運行番号 1/2 は，午前中に車庫Aを出発して昼前に車庫Aに戻る運行と，午後に車庫Aを出発して夜にB駅に停泊する二回の運行で構成される。運行番号 3 は午前中にB駅を出発して夜に車庫 A に戻る運行である。運行番号 4 は昼過ぎまで車庫Aにいて夜は車庫 A に戻る運行を表している。

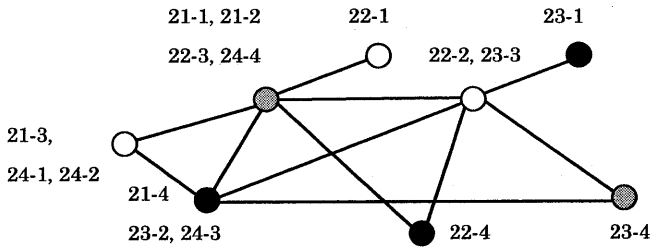
線区が所有する車両編成に対して，どの車両編成がどの運行を担当するかを日ごとに割り付けたスケジュールが車両運用計画である。先の例の運行を 3 つの車両編成に割り付けた四日間分の車両運用計画表の例を次にに示す。

編成番号	21(月)	22(火)	23(水)	24(木)
#1	1　2	3　→	予備　4	1　2
#2	3　→	予備　4	1　2	3　→
#3	予備　4	1　2	3　→	予備　4

一般的には，このように各運行を各車両編成がローテーションを組んで担当する。列車運行表は，翌日の運行を一段落としで順次ローテーションできるように作られている。しかしながら，一定距離を走行するごとに検査運用が入るために，実際の運用計画は完全なローテーションにはなっていない。この運用計画において，21 日に事故等が発生したために途中駅での車両交換などが発生して，車両運用の変更が発生したと考える。例えば、下図のような 21 日の運用実績になったとすると 22 日以降の運用計画では支障が発生する。編成#1 は 21 日の夜に車庫 A で停泊するが，22 日の予定が運用 3 であり，この運用は駅Bから出発する運用となっている。したがって，22 日以降の運用計画を修正する必要が生じる。この修正を行う問題が車両運用計画修正問題である。

編成番号	21(月)	22(火)	23(水)	24(木)
#1	3 →	3 →	予備 4	1 2
#2	予備 4	予備 4	1 2	3 →
#3	1 2	1 2	3 →	予備 4

この問題を上記のグラフの色分け手法で解くにはまず計画を元に戻す目標日を決めそこまでの探索範囲をグラフにした後に編成の数で色分けを行えばよい。詳細は省くが上の例で元の予定に戻す目標日を 24 日の午前とした場合のグラフを 3 色に塗り分けた例を以下に示す。このグラフの色分けを元に車両運行表を作成すれば、24 日には元の運用に戻る修正された計画が得られる。



3.2. グラフの縮約の必要性

グラフカラーリングには LSM を用いたが、実問題においてはジョブ間の制約を表現したグラフを工夫なしで色分けを行うことが難しいことが分かった。これは、実問題から作成されるグラフでは、グラフ内にリソース数と同じ数のノードを持つクリークが多数、しかも、それらが互いに密に隣接して発生することと関連する。LSM では制約違反を起こしているノードの色を変更して制約違反数を減らしていく手法であり、この場合、探索空間の構造が解の周辺で台地状になっているため、うまく解を見つけることができなくなることが原因である。そこで、問題固有の性質を利用してグラフの構造を予め縮約し、解空間の構造を変える事で色分けを行えるようにした。

実験に用いた路線の場合は,「車両交換ができるのは車庫だけであり、車庫は路線上一つだけである。」という条件がある。この場合、車庫を出発して半日で車庫に戻る半日運用は，グラフカラーリングの段階ではノードを省略しておいて，後で車両への割り付けを行う事が容易である。カラーリング後に半日運用に対して車両を割り付けるときは，元の計画どおりの車両が使えるかどうかを判断し，それ以外の場合は余っている車両を適当に割り付けれるだけで制約条件は満たされる。先の例では 22-1，22-4，23-1，23-4 がカラーリングの段階で省略できる半日運用である。この省略で，カラーリングは非常に簡単になる。実際の問題でもこの縮約を行う事で LSM によるグラフカラーリングが可能になった。

4. 実験と検証

実験システムをパソコン上に組み、実問題規模の実験を行った。紙面の都合があるので、修正後の結果のみを示す。縦軸にリソースである列車編成番号をとり、横方向に半日単位で運用を記入してある。網掛けの部分が元の計画と異なる運用の部分である。この例は 9 日に網掛けの部分の運用に変更があった例で、17 日の午後まで計画の変更が必要と思われていた例である。自動作成した結果では計画の乱れは 17 日の

午前に留まり、半日早く復旧できることがわかった。また、変更のなかった編成への影響も少なく押さえられている事も確認できる。なお、「検査」、「整備」、「試験」、「回送」と書かれた運用は車両点検に関係のある運用であり、計画の変更は行えない運用である。また、「留置」と書かれた運用は車両基地以外の駅の引き込み線での滞泊になる。これらを含んだ計画に対しても本手法の範囲で対応が可能である。本結果を得るのにかかる時間は CPU が Pentium100MHz、主記憶 48MB のパソコンでも数秒である。

編成	9 火		10 水		11 木		12 金		13 土		14 日		15 祝		16 火		17 水		18 木	
1	15	..	55	55	19	19	45	..	11	..	43	..	07	..	29	29	35	..	17	..
2	53	57	05	05	51	51	03	..	09	..	47	47	5	5	41	41	31	予備	15	..
3	57	21	57	57	07	43	15	..	51	..	予備	予備	予備	予備	予備	27	43	27	43	43
4	35	..	17	..	53	53	11	..	43	..	07	..	29	..	35	..	17	..	53	53
5	23	留置	29	29	35	..	17	..	49	..	予備	R15	01	..	25	25	39	..	37	37
6	41	17	53	予備	17	..	53	53	予備	予備	予備	R19	51	..	21	21	57	57	07	07
7	43	43	09	09	41	41	31	予備	予備	13	49	..	予備	R15	01	..	25	25	39	..
8	45	..	13	検査	21	21	57	57	予備	15	01	..	21	21	39	..	37	37	49	留置
9	25	37	49	留置	33	..	27	07	01	..	21	予備	予備	予備	17	..	53	53	予備	回送
10	05	03	07	51	15	..	55	55	予備	R17	39	..	09	..	51	51	03	..	11	..
11	19	19	45	..	5	5	51	51	03	..	53	予備	予備	R19	55	55	19	19	45	..
12	31	予備	15	..	55	55	19	回送	整備	整備	整備	整備	整備	整備	整備	回送	予備	予備	21	21
13	13	予備	21	21	57	57	09	09	33	..	25	21	留置	37	37	37	49	留置	33	..
14	51	55	03	..	03	..	13	検査	07	..	29	..	31	..	09	09	41	41	31	予備
15	予備	予備	予備	予備	予備	予備	21	21	53	予備	予備	予備	予備	予備	23	留置	29	29	35	..
16	01	..	25	25	39	..	37	予備	予備	予備	予備	R17	39	..	05	05	51	51	03	..
17	09	41	41	07	01	..	25	25	留置	37	留置	35	45	留置	33	..	27	07	01	..
18	37	09	41	41	31	予備	予備	27	41	予備	予備	予備	11	..	47	..	13	検査	13	検査
19	21	25	39	39	37	07	01	..	21	21	留置	37	留置	35	49	留置	33	..	27	27
20	03	07	01	..	25	25	39	..	留置	35	45	留置	27	..	27	予備	21	21	57	57
21	11	11	11	..	47	..	07	37	45	留置	27	..	23	..	43	19	45	..	05	05
22	33	..	27	27	43	37	49	..	27	..	23	..	41	13	53	53	23	留置	29	29
23	07	11	47	..	23	留置	29	29	31	..	03	..	53	予備	予備	43	07	43	09	09
24	29	29	35	..	11	..	47	..	05	05	33	..	25	予備	15	..	55	55	19	19
25	55	51	19	19	45	..	05	05	47	47	05	05	33	..	31	予備	15	..	55	55
26	39	..	37	37	49	留置	33	..	23	..	41	予備	予備	予備	予備	予備	予備	予備	予備	予備
27	不	予備	不	予備	試験	予備	予備	19	39	..	09	..	47	47	03	..	11	..	47	..
28	01	05	51	53	13	予備	23	..	29	..	31	..	03	..	57	57	09	09	41	41
29	27	27	43	43	09	09	41	41	25	予備	予備	13	48	..	07	07	01	..	25	25
30	47	..	23	留置	29	29	35	..	予備	予備	11	..	43	..	11	..	47	..	23	留置
31	49	留置	33	..	27	27	43	43	予備	19	51	..	予備	R17	45	..	05	05	51	51

車

両運用計画の修正結果(自動修正による)

5. まとめ

グラフカラーリングの手法である LSM を用いて車両の運用計画を自動修正するシステムを試作し，実問題規模の実験による検証を行った。この結果，元の計画に出来るだけ近い計画を作成するという点で良い修正結果を短時間に得ることができることがわかった。一般的にリソース割り付け問題に対しては今回と同じ枠組みが利用できる可能性がある。したがって，非常に有望な最適化方式であると考えられる。

参考文献

[1] 西岡 靖之: 状況対応型スケジューリングシステムにおけるペナルティ伝播グラフを用いた計画修正方法，経営システム 5, 263-271, (1995)

[2] 西岡 靖之: 状況対応型スケジューリング問題へのアプローチ，計測と制御 33, 571-574, (1994)

[3] A. Yoshikawa, K. Fujisawa, S. Morito and M.Kubo: An Application of the Life Span Method to the Graph Coloring Problem, 統計数理研究所共同リポート 53「最適化：モデリングとアルゴリズム 4」, 105-137, (1994)