Vol.2012-MPS-87 No.24
2012/3/2

移動体の通過順序付けにおける タブーサーチへの長期記憶の導入結果

澤田めぐみ† 白石將† 尾崎敦夫† 松村寛夫††

複数の移動体の通過順序付けには，一般に先着順方式が採用されることが多い．しかし，先着順方式を適用する場合，遅延伝搬により，交通全体として大きな遅延が発生する危険性がある．そこで，先着順からの順序入替えを許容し，最適化手法を用いて遅延を抑制する順序入替え方式を提案している．提案方式では，順序付けの決定に，最適化手法の１つであるタブーサーチ(TS:Tabu Search)を用いている．本稿では，提案方式で利用する TS に長期記憶を導入した．評価の結果，提案方式で利用する TS の性能向上には，長期記憶を用いるよりも，最良解が一定期間未更新となった場合に探索を初期解（先着順の順列）へ戻して再スタートする方が良いことが確認された．

Introduction of Long Term Memory into Tabu Search for Object Ordering

Megumi Sawada†, Masashi Shiraishi†, Atsuo Ozaki† and Nobuo Matsumura ††

In a transportation system, Moving objects passing through a common region should be ordered at the entrance. Usually, a first-come-first-served (FCFS) method is applied. However, FCFS could cause delay propagation among these objects, which could result in substantial total delay. To solve this problem, we proposed an ordering method that minimizes the total sum of delays considering fairness. The proposed method determines the order by using a tabu search (TS) that is a optimization algorithm. In this paper, we introduce a long term memory into the TS for the proposed method. Experimental results show that a restart of the search (the search is restarted from the FCFS order when the best solution has not been updated in a given number of iterations) provides better performance than the long term memory for the proposed method.

1. はじめに

複数の移動体が関与する交通システムでは，同一領域を移動体間で共有して通過することが多い．このような領域への入域点では，移動体間の通過順序付けが必要であり，一般に先着順方式が採用されることが多い．先着順方式は，移動体間の公平性が確保できるとの利点がある．しかし，先着順方式を適用する場合，先行する移動体に遅延が発生すると後続の移動体に遅延が伝播し，交通全体として大きな遅延が発生する危険性がある．そこで，公平性を大きく損なわない範囲内で先着順からの順序入替えを許容し，総遅延時間（全体の遅延）を最小化する順序入替え方式を提案している[1]. 提案方式では，順序付けの決定に，最適化手法の１つであるタブーサーチ(TS:Tabu Search)[2]を用いている．TS は，解を一つ保持し，その解を少し変化させた近傍解集合の最良解を選択することを繰り返すことで評価値の良い解の探索を行う.その際に，探索が後戻りして同じ解ばかりを探索すること（サイクリングと呼ばれる）を防ぐため，選択した最近の解の生成方法に関する情報（タブー属性）をタブーリスト（短期記憶と呼ばれる）に記憶し，一定期間，その属性を持つ解の選択を禁止する．TS ではこのようにして，局所最適解に陥ることを防ぎ，探索の多様化を図る．提案方式を簡単なシナリオで評価したところ，先着順方式と比べ，公平性を大きく損なわない範囲で総遅延時間を約 20～50%に抑制する効果があることを確認した[1].

本稿では，提案方式で利用する TS の性能向上を図るべく，TS の代表的な改良手法である長期記憶[2]を導入した結果について述べる．

2. 想定する交通モデル

想定する交通モデルについて，共有領域，移動体の順に説明する（図 1 参照）.

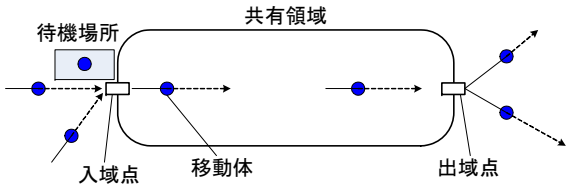


図 1 想定する交通モデル

† 三菱電機株式会社 情報技術総合研究所
Information Technology R&D Center, Mitsubishi Electric Corp
†† 三菱電機株式会社 インフォメーションシステム事業推進本部
Information Systems & Network Service Group, Mitsubishi Electric Corp

2.1 共有領域のモデル

- 複数の移動体が存在し，これらが共通に利用する単一の共有領域が存在する．
- 共有領域には入域点および出域点がそれぞれ 1 つずつ存在する．各移動体は入域点から入り，共有領域内を入域点から出域点に移動後，出域点から出てくるものとする．
- 共有領域の入域点直前に移動体の待機場所が存在し，ある定められた待機時間か，その整数倍の時間だけ，そこで待機可能である．ここで，待機場所を同時に利用する移動体数に制限はないものとする．

2.2 移動体のモデル

- 各移動体が共有領域の入域点に到着する時刻は予め決まっているが，待機場所を利用することにより，前後の移動体との間の間隔の調整や，共有領域に進入する順序の入れ替えが可能である．
- 移動体はある程度の速度調整が可能であり，共有領域の入域点から出域点への移動に要する時間は，最短移動時間～ 最長移動時間の範囲で自由に設定できるものとする．
- 安全性確保のため，連続する移動体の前後に所定の規定間隔が空いていることが必要である．本稿では簡単のため，出域点の通過時刻の時間間隔が所定の閾値以上であれば，規定間隔の制約を満たしているものと見なす．
- 各移動体の遅延時間を，出域点通過時刻に基づいて定める．その際，他移動体が存在しないと仮定した場合の最早の出域点通過時刻を基準とする．従って，移動体の遅延時間は，全て 0 以上の値となる．

上記の条件の下で，各移動体を，可能な限り小さな遅延で移動するよう経路生成（スケジューリング）することが，本稿の目的である．この際，所定の規定間隔を満たすため，以下の手段を利用する．

- 共有領域における速度調整．
- 待機場所の利用．

以上より，移動体スケジュールは個々の移動体の待機場所の利用状況および，(速度調整後の)出域点通過時刻で定まる．

3. 順序入替え方式

順序入替え方式（提案方式）は，移動体の共有領域への入域順序を，待機場所を利用して先着順から入れ替えることにより，遅延伝播を解消し，総遅延時間の最小化を図る方式である．この先着順入替えには最適化手法である TS を利用する．まず，提案方式の処理の流れを述べた後に，提案方式で利用する TS への長期記憶の導入について述べる．

3.1 処理の流れ

提案方式の処理の流れを図 2 に示す．

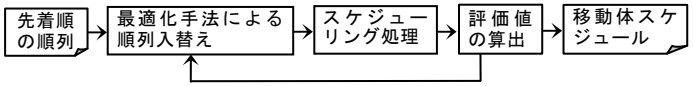


図 2 順序入替え方式の処理の流れ

提案方式では，以下の手順①～③で構成されるループ処理を規定回数だけ反復し，得られたスケジュールの中で評価値が最も良いものを最適として採用する．

① 「移動体の順列」および対応する評価値に基づき，最適化手法を用いて，新たな順列を複数生成する．

② 最適化手法が出力する各順列に関し，スケジューラが先頭の移動体から順に 1 台ずつ出域時刻を計算し，その時刻を時間軸上に割り当てていく．具体的には， 以下の時刻範囲[A]および[B]に共通に含まれる時刻範囲のうち，最早の時刻を，スケジューリング対象とする移動体の出域点通過時刻として決定する．

[A] 規定間隔を満たす出域点通過時刻範囲．具体的には，「既にスケジューリングされた移動体の出域点通過時刻±規定間隔」以外の時刻範囲．
[B] スケジューリング対象とする移動体が，出域点を通過可能な時間範囲．

スケジューラによる移動体の出域点通過時刻決定に関し，以下の状況下における 3 通りの例を図 3 (a)～(c)に示す．

- 最適化手法 が出力した移動体の順列が「…, A, … , B, … , P,…」である（順列が「…, B, … , A, … , P, …」でも良い）．
- 移動体 A, B が既にスケジューリングされ，移動体 P をスケジューリングする状況に関する．この際，P の最初の出域点通過可能時刻範囲が，A および B の出域点通過時刻付近であるものとする．

ここで，図 3 において，(c)は，待機場所の利用が必要な場合に対応する．

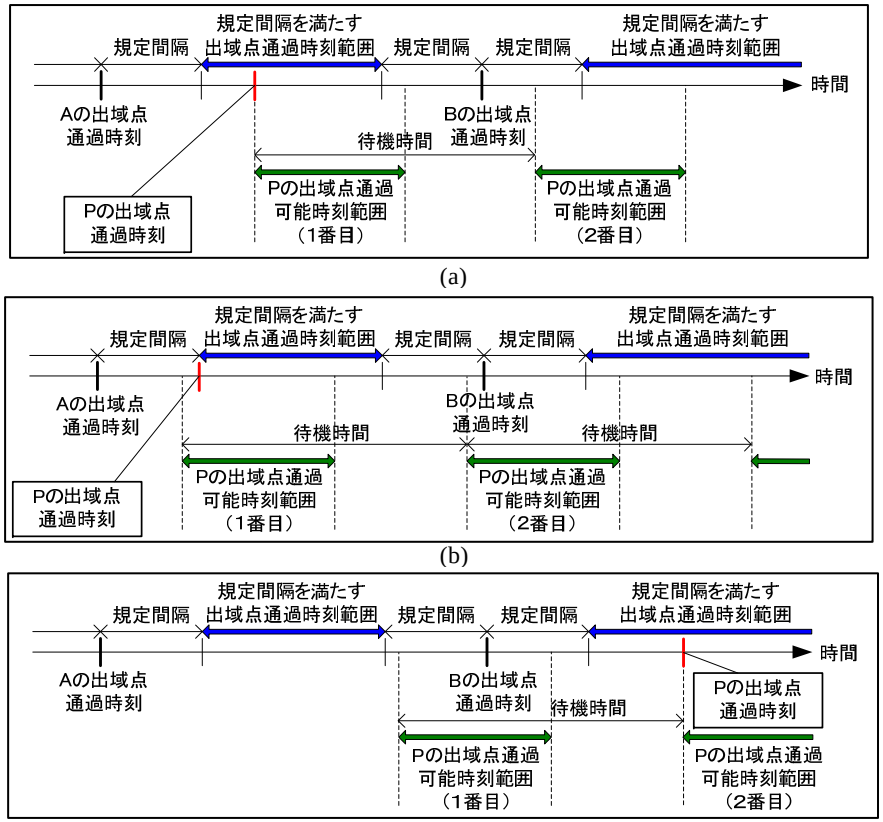


図 3　順序入替え方式のスケジューリング例

③ 生成された複数の移動体スケジュールのそれぞれの評価値を算出する．移動体スケジュールの評価値は，公平性も考慮した総遅延時間の最小化を行うために，式(1)に示す評価関数を利用して算出する．

$$F = (\text{総遅延時間}) + \alpha \times (\text{公平性の損失}) \qquad (1)$$

- 総遅延時間：各移動体の遅延時間の総和
- 公平性の損失：先着順から最も順序が遅れた移動体の順序差
- α：最適化項目トレードオフ調整用パラメータ

式(1)における公平性の損失は「先着順から最も順序が遅れた移動体の順序差」として算出する．順序差とは，「先着順方式での出域順序と提案方式での出域順序の差」を示す（例えば，先着順方式で生成された移動体スケジュールでは 2 番目に出域予定であった移動体が，提案方式で生成された移動体スケジュールでは 5 番目に出域予定となった場合は，順序差は 3 となる）．また，式(1)に記載されている α は，最適化項目のトレードオフ調整用パラメータである．この α の値を大きくするほど，公平性を保つことを考慮する度合いが大きくなる．また，α の値が 0 の場合は，総遅延時間のみを考慮した単一指標の最適化となる（評価結果は，文献[1]を参照）．

3.2 TS への長期記憶の導入

文献[1]において，提案方式で用いた TS は，短期記憶のみを利用していた．実装の内容を以下に示す．

- 解表現：移動体の順列 (スケジューリング順序を示す).
- 初期解：先着順の順列.
- 近傍解の生成法：現在解のスケジューリング順序をランダムに 1 組交換することで生成する．図 4 に，近傍解の生成例を示す．図 4 の場合，近傍解は，現在解のスケジューリング順序の 2 番目（a）と 5 番目（b）を交換することで生成されている．本稿では，現在解の全ての近傍解を生成するのではなく，予め設定した数の近傍解をランダムに生成する実装を利用する．

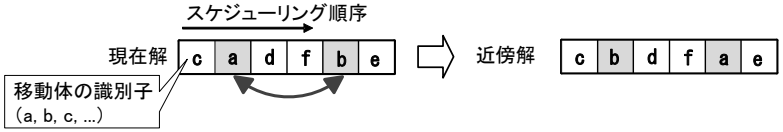


図 4　近傍解の生成方法

- 短期記憶への記憶内容：現在解を更新する（次の解へ遷移する）際に選択した「スケジューリング順序の交換位置の組」を短期記憶に登録する．例えば，図 4 の近傍解で現在解を更新した場合，「スケジューリング順序の 2 番目と 5 番目の組」を短期記憶に登録する．短期記憶に登録した交換位置の組は，登録してから一定期間（ループ回数で指定され，短期記憶長と呼ばれる）が経過すると破棄される．
- 探索時の短期記憶の利用方法：現在解を更新する際に，「短期記憶に登録されている交換位置の組」によって生成された近傍解（以後，タブー解と呼ぶ）での更新を禁止する．例えば，図 4 の近傍解で現在解を更新した場合，それから一定期間は，スケジューリング順序の 2 番目と 5 番目の交換によって生成される近傍解での更新は禁止される．ただし，タブー解が「今まで探索した解の最良解の評価値」よりも良い評価値である場合，タブー解での更新も行うものとする．

本稿では TS の代表的な改良手法である長期記憶（頻度メモリ）[2]を提案方式の TS に導入する．長期記憶も短期記憶と同様に，探索の多様化を目的としている．両者の

違いを以下に示す.

- 短期記憶の利用目的 : 直前の解の変更操作等を記憶し，直前に探索した解への遷移を禁止することで，「局所最適解から脱出し，かつ，探索のサイクリングを防ぐこと」.
- 長期記憶の利用目的 : 解の特定の変数を変更した回数等を記憶し，類似した変更操作が頻繁に行われることを防ぐことで，「探索を未探索領域へ方向づけること」.

具体的な長期記憶の利用方法としては，解の評価関数に，長期記憶に基づくペナルティ項を加え，頻繁に選択されている解の変更操作による現在解の更新をしにくくする等が挙げられる．しかし，この際に，ペナルティが大きすぎると，解の探索性能が悪くなるといわれており，以下の方法がとられる[2].

- ペナルティ項に重み係数を付け，ペナルティ項の値を調整する.
- 探索の多様化が必要と判定される場合にのみペナルティ項を加える．具体的には，局所最適解から脱出を図る場合（近傍解集合の中に現在解よりも良い解がなく，現在解と比べ，改悪である解に探索を進める場合）や最良解が一定期間未更新である場合等に，多様化が必要と判定する.

本稿では，以下に示す長期記憶の内容に対して，探索時の長期記憶の利用方法が異なる(ア)～(ウ)をそれぞれ実装した.

- 長期記憶への記憶内容 : 現在解の更新において「スケジューリング順序の交換位置の各組合せ」が選択された回数．この回数のカウントは，全探索期間で行い，かつ，途中でリセットしないものとする.
- 探索時の長期記憶の利用方法 (ア):探索において局所最適解から脱出を図る場合に，長期記憶を参照し，頻繁に選択されている現在解の更新内容（順序の交換位置の組）にペナルティを与える．具体的には，解の評価値を式(2)に示す評価関数を利用して算出する.

$$F' = F + \beta \times Freq(i, j) \quad (2)$$

F：移動体スケジュールの評価値（式(1)の値）
Freq(i, j)：現在解の更新で交換(i, j)が選択された回数
β ：ペナルティ値調整用パラメータ

- 探索時の長期記憶の利用方法 (イ):探索において局所最適解から脱出を図る場合，かつ，最良解が一定回数未更新となった場合に，長期記憶を参照し，頻繁に選択されている現在解の更新内容にペナルティを与える．このペナルティは，(ア)の場合と同様に，式(2)を用いて与える.
- 探索時の長期記憶の利用方法 (ウ):最良解が一定回数未更新となった場合に初期解（先着順の順列）から探索を再スタートする（この再スタートは初回 1 回のみ実施する）．この再スタート後の探索において，長期記憶を参照し，頻繁に選択されている現在解の更新内容にペナルティを与える．このペナルティは，(ア)の場合と同様に，式(2)を用いて与える.

4. 評価

評価条件ついて説明する．そして，評価結果を示し，考察する.

4.1 評価条件

まず, 評価シナリオを以下に示す. ここで, 時間は任意の単位における比率とする.

- 移動体は全部で 60 台であり，入域点に時間間隔 90 で到達する.
- 移動体の種類は A, B の 2 種類であり, 移動体 A, B の入域点への到達パターンは，ランダムに生成された図 5 に示すパターンである.
- 出域点で要請される移動体間の規定時間間隔は，連続する移動体の種類に応じて定まるものとし，表 1 に示す 2 つのシナリオを設定した（必要な規定間隔が大きいほど，移動体間で遅延が伝播し易い状況となる）
- 待機時間は，1 回の待機あたり 240 とする.
- 移動体が共有領域の移動に要する時間は 830～920 の範囲で調整可能とする.

入域点到達順序	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
移動体種類	A	A	A	A	A	A	A	B	B	A	B	A	B	A	A	B	B	B	B	A	B	A	A	B	A	B	A	B	B	B

入域点到達順序	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
移動体種類	B	B	B	B	A	B	A	B	A	A	B	B	A	A	B	A	B	A	A	B	B	A	A	B	B	B	A	B	B	B

図 5　2 種類の移動体 A, B の入域点到達パターン

表 1　移動体の種類に応じた出域点での規定時間間隔

前方の移動体	後続する移動体	規定時間間隔	
		シナリオ 1	シナリオ 2
A	A	120	100
A	B	120	150
B	A	120	50
B	B	120	50

評価において，比較対象とした方式を表 2 に示す．TS のパラメータ設定に関し，以下は全評価で共通とする.

- 最適化のループ処理の終了条件：1000 回.
- 1 ループあたりの近傍解の生成数：400 個.
- 短期記憶長：20.

表 2　評価で比較対象とした方式

TS_Short	短期記憶のみを利用する.
TS_Long(Local)	短期記憶と長期記憶の実装(ア)（β=10）を利用する.
TS_Long(Conv)	短期記憶と長期記憶の実装(イ)（長期記憶利用開始までの最良解の未更新回数 200 回，β=20）を利用する.
TS_Long(Reset)	短期記憶と長期記憶の実装(ウ)（探索の再スタート実施までの最良解の未更新回数 200 回，β=50）を利用する.
TS_Reset	TS_Long(Reset)においてβ=0 としたもの（つまり，長期記憶が未使用）.

また，提案方式において公平性の考慮度合いを調整するパラメータであるαは，「0 / 100 / 1000 / 5000 / 10000」に設定して実行した.

4.2 評価結果

表 2 に示す方式をそれぞれ 200 試行実行し，各試行で得た解の評価値（式(1)で定義した移動体スケジュールの評価値）について，標準偏差を表 3（シナリオ 1），表 4（シナリオ 2）に示し，分布を図 6（シナリオ 1），図 7（シナリオ 2）に示す.

表 3，表 4，図 6，図 7 より，以下[A]～[D]のことが確認された.

[A] TS_Long(Local), TS_Long(Conv)は，TS_Short と優劣がつけ難い結果となった.

[B] TS_Long(Reset), TS_Reset は，TS_Short と比べ，得られる解の評価値の平均は同程度であるが，分散を減少する効果が確認された.

[C] TS_Long(Reset), TS_Reset が，得られる解の評価値の分散を減少する効果は，シナリオ 1 と比べ，シナリオ 2 の方が大きかった.

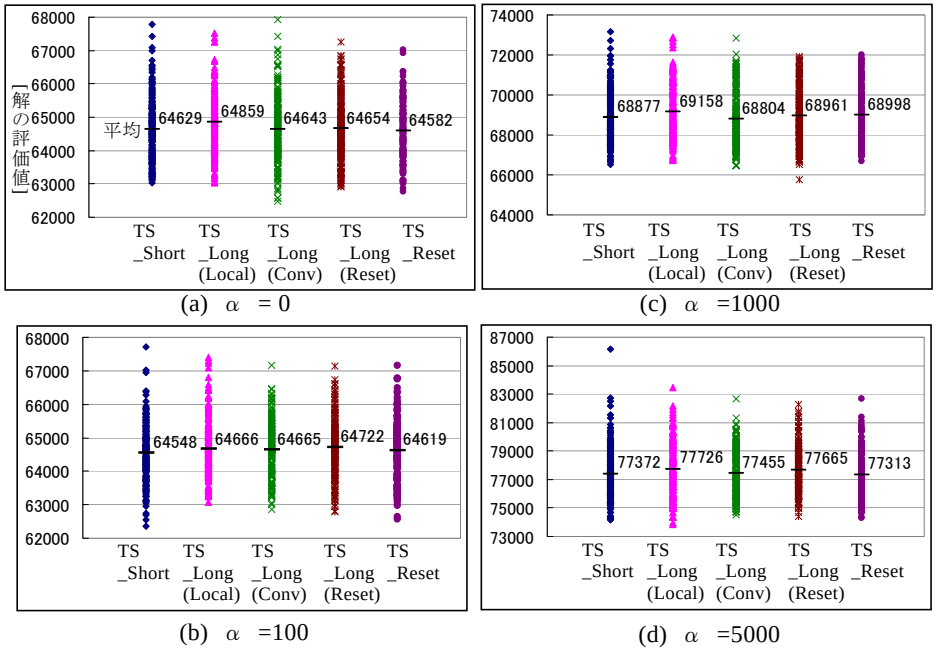
[D] TS_Long(Reset)と TS_Reset は優劣つけ難い結果となった.

表 3　シナリオ 1 における TS_Short, TS_Long(Local), TS_Long(Converge), TS_Long(Reset), TS_Reset から得られた解の評価値 F（式(1)）の標準偏差

	TS_Short の標準偏差	TS_Short に対する標準偏差の増減率[%]			
		TS_Long (Local)	TS_Long (Conv)	TS_Long (Reset)	TS_Reset
α＝0	853	0	＋16	＋2	＋2
α＝100	930	－7	－10	－4	－9
α＝1000	1363	0	＋3	－2	－6
α＝5000	1875	＋2	－9	－12	－11

表 4　シナリオ 2 における TS_Short, TS_Long(Local), TS_Long(Converge), TS_Long(Reset), TS_Reset から得られた解の評価値 F（式(1)）の標準偏差

	TS_Short の標準偏差	TS_Short の標準偏差に対する増減率[%]			
		TS_Long (Local)	TS_Long (Conv)	TS_Long (Reset)	TS_Reset
α＝0	567	－8	－1	－35	－32
α＝100	656	＋2	＋2	－31	－26
α＝1000	877	－6	－7	－28	－23
α＝5000	967	0	－2	－13	－19



(a) α = 0　(c) α =1000

(b) α =100　(d) α =5000

図 6　シナリオ 1 における TS_Short, TS_Long(Local), TS_Long(Converge), TS_Long(Reset), TS_Reset から得られた解の評価値 F（式(1)）の分布

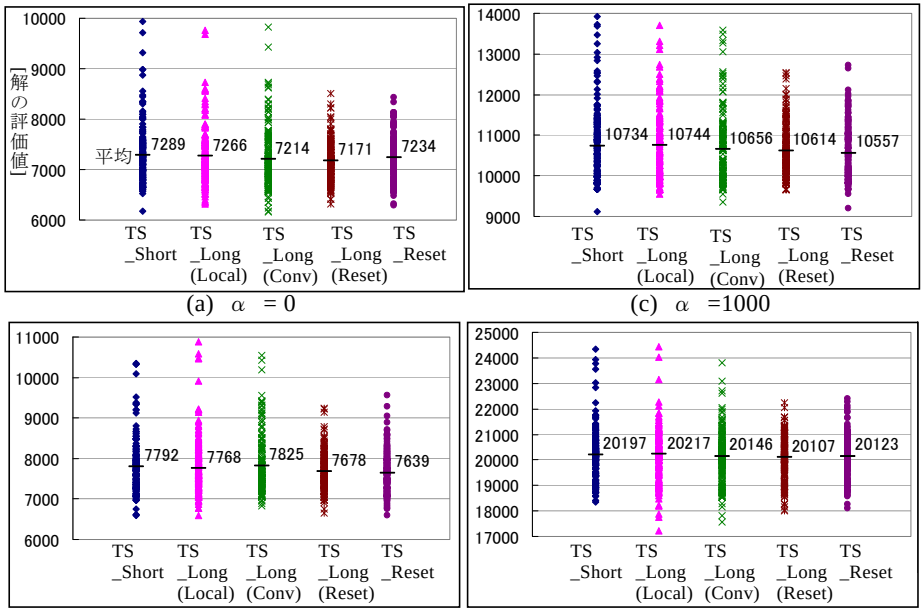


図 7　シナリオ 1 における TS_Short, TS_Long(Local), TS_Long(Converge), TS_Long(Reset), TS_Reset から得られた解の評価値 F（式(1)）の分布

4.3 考察

本節では，評価結果に関する考察を述べる．まず，評価結果[A][B]より，TS の探索性能の向上には，長期記憶を用いるよりも，最良解が一定期間未更新となった場合に探索を初期解（先着順の順列）へ戻し，再スタートする方が良いことを確認した．これは，図 8（TS_Short から得た最良解に対応する移動体のスケジューリング順序）から確認されるように，本問題の性質として，先着順に近い順序を持つ解が良い評価値を持つ傾向があり，長期記憶による多様化では，先着順と大きく異なる局所最適解へ陥った場合に，局所最適解から脱出し，かつ，先着順に近い解へ探索を進めることが難しいためであると考える．

次に，評価結果[C]について述べる．表 5 に，TS_Short における探索の収束状況（最良解を得た探索のループ回数）を示す． 表 5 より，シナリオ 1 よりもシナリオ 2 の方が探索の収束状況が早い傾向があることが確認された．このことから，探索の収束状況が遅い傾向があるシナリオ 1 では，再スタート後の最適化のループ回数が短いケースが多いため，再スタートによる探索性能の向上が小さかったと考える．

最後に，評価結果[D]について述べる．現在解の全近傍解数（順序交換位置の組合せ数）が 1770 であるのに対して，各ループ処理で生成する近傍解数は 400 である．よって，よって，再スタート後の探索では，長期記憶を利用しなくても再スタート前と同じ探索（再スタート前と同じ順序で同じ解を選択し続けること）が行われることは少なく，長期記憶を利用することによる探索性能の向上が確認されなかったと考える．

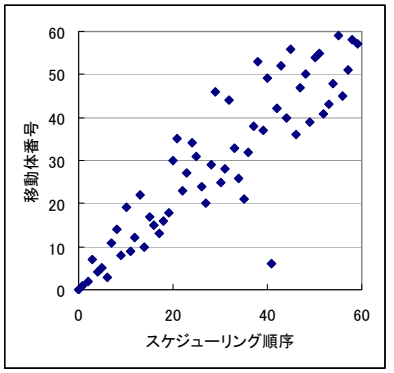


図 8　シナリオ 1，α =100 において TS_Short から得た最良解に対応する移動体のスケジューリング順序（移動体番号は先着順に付けられた番号を示す）

表 5　TS_Short における最良解を得た探索のループ回数（200 施行の平均値）

α	シナリオ 1	シナリオ 2
0	914	554
100	886	534
1000	826	665
5000	650	667

5. むすび

移動体の順序入替え方式において順序決定に利用する TS の探索性能の向上には，長期記憶を用いるよりも，最良解が一定期間未更新となった場合に探索を初期解（先着順の順列）へ戻して再スタートする方が，得られる解の評価値の平均は同程度であるが，解の分散を減少する効果があることが確認された．

参考文献

[1] 澤田めぐみ，白石將，尾崎敦夫，松村寛夫，“移動体の通過順序付けにおける最適化手法の比較評価，” FIT2011 講演論文集, P217-218(第 1 分冊), 2011.
[2] 柳浦陸憲，茨木俊秀: 組合せ最適化―メタ戦略を中心として―，朝倉書店, 2000.