GA によるヒューリスティック探索の最適化
— バス仕業ダイヤの作成 —

仙石浩明 † 吉原郁夫 † 今川徹三 ††

sengoku@sdl.hitachi.co.jp

(株) 日立製作所 システム開発研究所 † 情報システム事業部 ††

〒 215 川崎市麻生区王禅寺 1099

遺伝的アルゴリズム (GA) は、探索が大域的、制約条件の変化に柔軟、などの長所があり、開発・保守工数を削減できるが、遺伝子コーディング法、交差方法などを問題毎に考案しなければならない。一方、従来から広く用いられてきたヒューリスティック探索法は、人手で解かれていたような問題を解く場合は、容易にアルゴリズムを作ることができる。しかし実問題に適用するには、より詳細な知識を組み込むなどのチューンアップが必須であり、開発・保守工数がかかるという短所がある。

そこで、ヒューリスティック探索において探索木の分枝選択に優先順位を定め、この優先順位を GA で最適化する手法を提案する。本提案手法をバス仕業ダイヤ作成システムに応用し、実用上十分な仕業ダイヤ作成が可能となった。本システムは実際にダイヤ改正で使われた。

GA based Optimization of Heuristic Search
—Bus Drivers Scheduling—

Hiroaki Sengoku†, Ikuo Yoshihara† and Tetsuzo Imagawa††

Systems Development Laboratory†, Information Systems Division††,

Hitachi Ltd.

1099 Ohzenji Asao, Kawasaki, Kanagawa 215 JAPAN

Genetic Algorithms (GAs) have advantages in ability to search globally and flexibly, but we have to design the gene coding and the crossover method depending on each problem. On the other hand heuristic search algorithms are easy to develop and have been widely used to solve the problems, which have been conventionally solved manually, but for practical applications, we have to tune up the algorithms, for example, adding knowledge in detail, adjusting parameters, and so on.

In this paper, we present a method using GA as an optimizer of the heuristic search. In our method, GA tunes the priorities of choosing branches in search tree, with which heuristic search algorithm can find optimal solutions. We apply our method to bus drivers scheduling systems. It can generate enough practical schedules, and is already used for revising drivers schedules.

1 はじめに

遺伝的アルゴリズム (GA)[3] は、生物進化の過程をモデル化した、解探索アルゴリズムである。探索に関する知識が不要なため開発工数が少ない、問題の諸条件が変化しても柔軟に適応できる、探索が大域的である、などの長所を持つ。

このような長所がある反面、GA は問題毎に遺伝子コーディング法および交差方法を考案しなければならない。コーディング法等の善し悪しによって探索効率は大きく左右されるが、どのようなコーディング法を用いれば効率向上が図れるかは問題に強く依存し、新しい問題に応用を試みるたびにかなりの労力が必要となるのが現状である。

一方、従来から広く用いられてきたヒューリスティック探索は、人手で解かれていたような問題に関しては、人手でどのように解いていたかを整理して、ルールあるいは手続きの形で表現することが出来れば、比較的容易にプログラムを開発することが出来る。しかし解き方に関する知識を全て抽出出来ることは稀で、試行錯誤によるチューニングを行わないと満足のいく解が得られないことが多い。また、ヒューリスティック探索は知識に基づいて探索木の枝刈りをするわけであるから、大局的な探索を行うことが難しい。

そこで本報告では、ヒューリスティック探索に GA を組み込み、両者の短所を補い、長所を活用する手法を提案する。

以下、2 章では、本報告であつかうヒューリスティック探索アルゴリズムの定義を行ない、このアルゴリズムに GA を組み込む手法を 3 章で提案する。4 章ではヒューリスティック探索によるバス仕業ダイヤプログラムに、提案手法に基づき GA を組み込んだ例を紹介する。

2 ヒューリスティック探索

従来、組合せ最適化問題の多くは、ヒューリスティックを用いた探索を使って解かれてきた。つまり、解空間のすべてを探索することは組合せ爆発により非現実的であるので、解がありそうな部分解空間だけを探索して、実用上十分な準最適解を見つける。どの部分空間に良い解がありそうかは経験に基づいて決められる。

本章では、本報告で扱う組合せ問題およびその解法であるヒューリスティック探索を定義する。

定義 1 M 個の互いに異なるアルファベットを $s_1, s_2, \ldots, s_M$ とする。このアルファベットの集合を Σ とおく。

互いに異なるアルファベットの並びを $\sigma = s_{i_1} s_{i_2} \cdots s_{i_m}$ $(m < M)$ とし、空列を λ とする。λ および、すべての σ からなる集合を Σ^* とする。 □

定義 2 (半順序) $\sigma = s_{i_1} s_{i_2} \ldots s_{i_m}$ と $\sigma' = s_{j_1} s_{j_2} \ldots s_{j_n}$ の間の半順序関係 $\prec$ を次のように定義する。

$$\sigma \prec \sigma' \iff m < n \ \wedge \ \forall k \leq m,\ s_{i_k} = s_{j_k}$$

□

$\forall \sigma \neq \lambda,\ \lambda \prec \sigma$ である。

定義 3 (組合せ最適化問題) 組合せ最適化問題を 3 つ組 $(\Sigma, \overline{f}, \underline{f})$ で表す。ただし、

- $\Sigma = \{s_1, s_2, \ldots, s_M\}$
- $\overline{f} : \Sigma^* \longrightarrow R^+$
- $\underline{f} : \Sigma^* \longrightarrow R^+$

である。R^+ は 0 または正の実数である。

$\overline{f}(\sigma_s),\ \underline{f}(\sigma_s)$ がともに定義される $\sigma_s \in \Sigma^*$ を「部分解」と呼ぶ。$\overline{f},\ \underline{f}$ は、すべての部分解 σ_s にたいし、次の条件を満たす。

$$\forall \sigma \in \Sigma^*,\quad \sigma \prec \sigma_s \ \Rightarrow \ \overline{f}(\sigma) \geq \overline{f}(\sigma_s)$$
$$\sigma \prec \sigma_s \ \Rightarrow \ \underline{f}(\sigma) \leq \underline{f}(\sigma_s)$$

また、$\overline{f},\ \underline{f}$ は、任意の部分解 σ_s にたいし、$\sigma_t \in \Sigma^*$ が存在して、$\sigma_s \prec \sigma_t \ \wedge \ \overline{f}(\sigma_t) = \underline{f}(\sigma_t)$ を満たす。σ_t を「解」と呼ぶ。また、任意の解 σ_t に対して $f(\sigma_t) \equiv \overline{f}(\sigma_t)$ を「評価関数」と呼ぶ。

以上の条件を満たす 3 つ組 $(\Sigma, \overline{f}, \underline{f})$ において、評価関数 $f(\sigma_t)$ を最小化する解 σ_t を求める問題を、組合せ最適化問題と定義する。 □

解 σ_t に対し、 $f(\sigma_t)$ を、 σ_t の「評価値」と呼ぶ。評価値が最小となる解を「最適解」と呼ぶ。

$\overline{f}(\sigma_s)$ は σ_s から生成可能な解の評価値の上界を与え、 $\underline{f}(\sigma_s)$ は下界を与える。どの程度良い上・下界を求めることが出来るかは問題に依存する。多くの問題において、現実的な計算量で上・下限を求めることは不可能であるから、上・下界の評価方法は多くの場合改善する余地があるが、本報告では一般的に知られている評価方法あるいは容易に考案可能な評価方法のみを考える。

十分に良い下界が得られる問題の場合は、ヒューリスティックを用いず、分枝限定法だけで効率良く解を求めることができる。しかし多くの問題、特に実問題においては解の生成の中途で、良い下界を求めることは困難であり、探索空間を限定するためのヒューリスティックが必要となる。

次に、組合せ最適化問題 $(\Sigma, \overline{f}, \underline{f})$ を解くヒューリスティック探索アルゴリズムを示す。この探索では、探索範囲を限定するため、最適解が得られるとは限らない。そこで限定された探索範囲における最も良い解を「最良解」と呼ぶことにする。

```
procedure main {
    x ← ∞
    search(Σ, λ)
}
procedure search(S, σ) {
    if f̄(σ) < x {
        x ← f̄(σ)
        if f̄(σ) = f̲(σ) then {
            σ_t = σ
            return
        }
    }
    S ← {s ∈ S | defined(f̲(σ·s))
            ∧defined(f̄(σ·s))
            ∧f̲(σ·s) < x}
    {T ⊂ S を選択} ……(*)
    for all s ∈ T do search(S − {s}, σ·s)
}
```

ただし、 x, σ_t は大局的変数、 '$\cdot$' は連接演算子である。 σ_t に最良解が返される。

上記アルゴリズムの $(*)$ で選択した T を、その探索局面における「選択枝集合」と呼ぶ。選択枝集合は、組合せ最適化問題の解の性質に関する知識を用いて求める。

ヒューリスティック探索の問題点は、選択枝集合を求める方法を、諸条件に応じてチューニングする必要があるという点である。選択枝集合の要素数が大き過ぎると組合せ爆発が起き、また小さ過ぎると探索が偏り、最適解が探索範囲から洩れるおそれがある。チューニングのための工数がかかる上、プログラムのロジックが複雑になるので、保守およびカスタマイズが困難になる。

3 GA によるヒューリスティック探索の最適化

前章で定義したヒューリスティック探索を GA により最適化する手法を提案する。最適化された探索で最良解を求めることにより、組合せ最適化問題 $(\Sigma, \overline{f}, \underline{f})$ の最良解を求めることができる。

本手法は、ヒューリスティック探索のアルファベットに全順序関係を導入し、この順序関係を GA で調整することによってヒューリスティック探索の最適化を図る。全順序関係を定めるために各アルファベットに正整数を割り当てるが、この正整数を以下、優先順位と呼ぶ。

アルゴリズムの概要を図 1 に示す。各個体の遺伝子列からアルファベットの優先順位を求め、ヒューリスティック探索において優先順位を参照しながら最良解を生成する。そしてその最良解の評価値を個体の評価値とする。

以下、アルゴリズムの詳細を説明する。

3.1 遺伝子表現法

遺伝子列は図 2 に示す、要素数 M の固定長ベクトルで、各要素 q_i は $1 \leq q_i \leq M - i + 1$ を満たす整数値である。このベクトルは、各アルファベットの優先順位を表す順序表現 [4] (Grefenstette による遺伝子表現法) である。

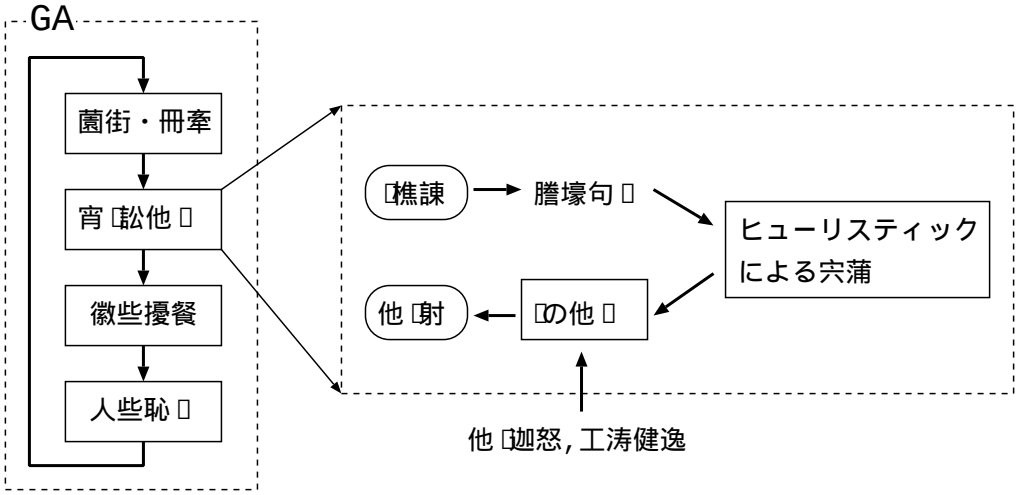

図 1: GA によるヒューリスティック探索の最適化

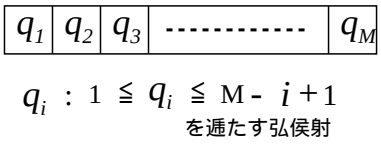

図 2: 遺伝子

遺伝子列 $\mathbf{q} = q_1 q_2 \cdots q_M$ からアルファベット s_i の優先順位 $p(s_i)$ を求めるアルゴリズムを次に示す。

$U \leftarrow \{1, 2, \ldots, M\}$
for $i \leftarrow 1$ **to** M **do** {
  $p(s_i) \equiv \begin{pmatrix} U \text{ の要素のうち、小さい方} \\ \text{から } q_i \text{ 番目の数字} \end{pmatrix}$
  $U \leftarrow U - \{p(s_i)\}$
}

あきらかに、優先順位 $p(s_i)$ $(i = 1, 2, \ldots, M)$ は互いに異なる 1 から M までの数値となる。

3.2 適応度評価

2 章で定義したヒューリスティック探索のアルゴリズムにおいて、最後の部分

$\{T \subset S$ を選択 $\} \cdots\cdots (*)$
for all $s \in T$ **do** search$(S - \{s\}, \sigma \cdot s)$
}

を次のように書き換えることにより、優先順位 $p(s_i)$ $(i = 1, 2, \ldots, M)$ に基づく探索を行なう。

$\{T \subset S$ を選択 $\} \cdots\cdots (*)$
$\{\forall s' \in T', \forall s \in (T - T'), p(s') < p(s)$ を
  満たす $T' \subset T$ を選択 $\} \cdots\cdots (**)$
for all $s \in T'$ **do** search$(S - \{s\}, \sigma \cdot s)$
}

$(**)$ において $|T'| \equiv 1$ とする。つまり探索木の枝分れがない探索である。組合せ爆発のおそれがないので、T の要素数が大きくても良い。すなわち選択枝集合を求める方法をチューニングする必要がない。

なお、GA による最適化が収束した後に $|T'| > 1$ とすれば、$|T'| = 1$ の場合に生成される解の近傍探索が可能である。

個体の遺伝子にコーディングされた優先順位に基づく探索によって得られる最良解の評価値をその個体の評価値とし、評価値の逆数を適応度とする。

3.3 自然淘汰

個体集合から R 個 (定数) の個体を取り除く。この時、個体集合の多様性を維持するため [1] に、次のように行う。まず適応度が高い順に個体を並べる。適応度が高い方から順に個体の適応度を調べ、直前の個体の適応度との差の絶対値が ϵ 以下である個体を全て取

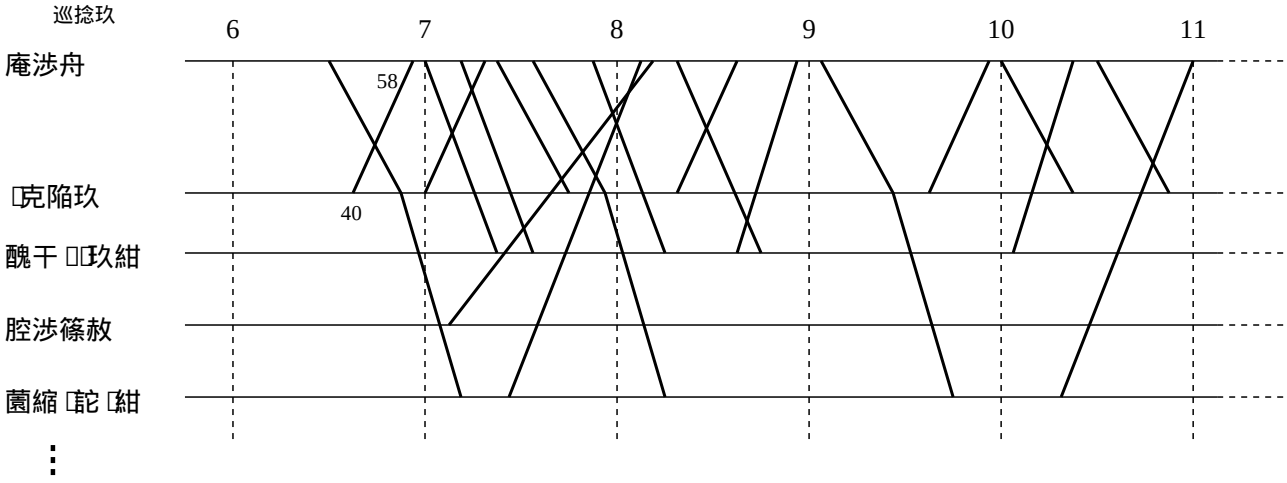

図 3: バスダイヤ

り除く。ただし取り除く個体数は R 個までとする。取り除いた個体数を r とする。$r < R$ の場合は、$R - r$ 個の個体を適応度が低い方から順に取り除く。

3.4 突然変異

最優良個体を除くすべての個体の遺伝子 q_i $(i = 1, 2, \ldots, M)$ の値を確率 P (定数) で、ランダムに選んだ値 $(1 \sim M - i + 1)$ に書き換える。一つ以上の遺伝子を書き換えた個体は、評価値および適応度を再計算する必要がある。

3.5 交差・増殖

交差は一点交差法を用いる。すなわち、親個体の遺伝子 $\mathbf{q} = q_1 q_2 \cdots q_M$, $\mathbf{q}' = q'_1 q'_2 \cdots q'_M$ に対して、$1 \leq r \leq M - 1$ を満たす乱数 r を生成し、子個体の遺伝子 $q_1 q_2 \cdots q_r q'_{r+1} q'_{r+2} \cdots q'_M$ を作る。遺伝子表現法に順序表現を用いているため、致死遺伝子が生じることはない。

ランダムに R 組の親個体を選んで交差を行ない、個体数を R 個増やす。自然淘汰において個体数が R 個減るので、世代間で個体数は一定である。

3.6 最適化の効率

親個体の表現型が子個体の表現型にどのくらい忠実に遺伝するかが、GA による最適化の効率を左右する [2]。本提案手法において、親個体の遺伝子 $\mathbf{q} = q_1 q_2 \cdots q_M$, $\mathbf{q}' = q'_1 q'_2 \cdots q'_M$ から交差によって生成される子個体の遺伝子 $q_1 q_2 \cdots q_r q'_{r+1} q'_{r+2} \cdots q'_M$ について考える。

まず $s_1, s_2, \ldots, s_r$ の優先順位は、明らかに親 $\mathbf{q}$ の優先順位を受け継ぐ。次に、$s_{r+1}, s_{r+2}, \ldots, s_M$ の優先順位は、

$$
\begin{aligned}
&U \leftarrow \{1, 2, \ldots, M\} - \{p | \exists i \leq r, p = p(s_i)\} \\
&\textbf{for } i \leftarrow r+1 \textbf{ to } M \textbf{ do } \{ \\
&\quad p(s_i) \equiv \begin{pmatrix} U \text{ の要素のうち、小さい方} \\ \text{から } q_i \text{ 番目の数字} \end{pmatrix} \\
&\quad U \leftarrow U - \{p(s_i)\} \\
&\}
\end{aligned}
$$

で求められるので、$p(s_{r+1}), p(s_{r+2}), \ldots, p(s_M)$ の間の大小関係は $p(s_1), p(s_2), \ldots, p(s_r)$ に依存しない。したがって親 $\mathbf{q}'$ の $s_{r+1}, s_{r+2}, \ldots, s_M$ の優先順位を受け継ぐ。

優先順位を受け継いだ子個体は、ヒューリスティック探索において、選択枝集合が同じであれば優先順位によって縮小した選択枝集合も親個体と同じになる。したがって子個体は親個体の表現型 (ヒューリスティック探索で生成される最良解) を受け継ぐ。

4 バス仕業ダイヤ作成への適用

4.1 仕業ダイヤ作成問題

図 3 のようなバス運行ダイヤが与えられた時、図中斜めの線分で表されたバスの運行に、運転手を割り

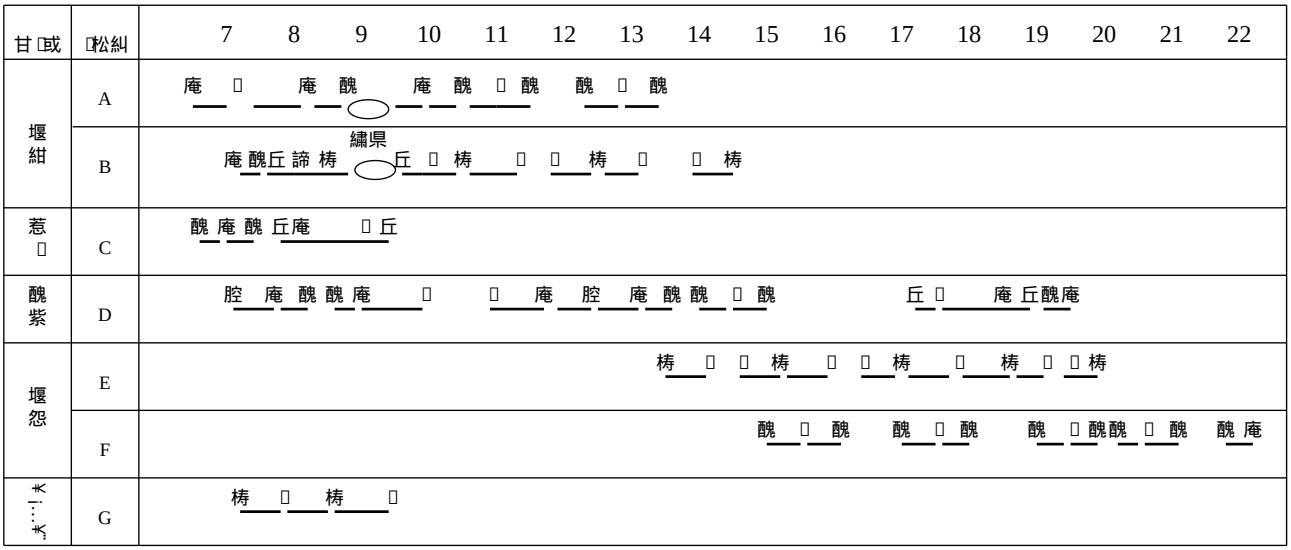

図 4: バス仕業ダイヤ

当てる仕業ダイヤ (図 4) を作成する問題が仕業ダイヤ作成問題である。

バスダイヤ (図 3) において、横軸は時間であり、バスの運行が斜めの直線で表されている。例えば、「及川支所」停留所を 6:40 に出発したバスは、「恵田町」停留所に 6:58 分に到着することが読みとれる。「及川支所」から「恵田町」までのこのような一続きのバスの運行を、山ダイヤと呼ぶ。

バス仕業ダイヤ (図 4) において、横軸は時間であり、山ダイヤが線分で表される。各行は運転手を表し、同じ行にある山ダイヤ全てを運転することを表す。例えば運転手「A」は、「恵 (恵田町)」停留所から出発して、「及」、「恵」を通って「長」に至り、8:44 から 9:30 までは朝食休憩を取り、 13:24 「長」まで乗務することを表している。各運転手の勤務を仕業と呼ぶ。

仕業ダイヤ作成は、従来ほとんどのバス会社において人手で行っていたが、我々が調査したバス会社では専門家が 3 ヶ月もかかるほど工数が大きい作業であり自動化が望まれていた。自動化によって工数削減および仕業ダイヤの質の改善が期待される。

仕業には例えば「午前」「中休」「長大」「午後」「パート」などの型があり、それぞれ勤務時間・運転時間の上限、食事休憩の有無などが定められている。さらに、運転が連続する場合、一定時間以内毎に休憩を取る必要がある。また、停留所によっては、一定時間以上の駐車が禁止されている所があり、駐車時間が長い場合は最寄りの営業所等に回送しなければならない。

以上のような制約を満たしつつ、回送本数が少なく、かつ各運転手の労働時間のバランスがとれるように山ダイヤを割り当てる必要がある。

4.2 GA による仕業ダイヤ作成

仕業ダイヤ作成を自動化するために、まず人手による仕業ダイヤ作成方法を参考にしながら、ヒューリスティック探索による仕業ダイヤ作成プログラムを作成した。このプログラムは、特定のバス会社に依存する知識は用いず、またチューニングも行なっていないので、選択枝集合を求める方法が不適切であり、すべての制約条件を満たすことは不可能であった。

このヒューリスティック探索プログラムでは、山ダイヤを時刻順に並べた山ダイヤリスト

$$Y = y_1 y_2 \cdots y_L \quad (y_i\text{: 山ダイヤ}, L\text{: 山ダイヤの数})$$

を定め、このリストの順に山ダイヤを一つずつ各運転手に割り当てることにより仕業ダイヤを作成する。山ダイヤの割当順が固定であることから、仕業ダイヤは

$$d_{j_1} d_{j_2} \cdots d_{j_L}$$

と表現できる。ただし d_{j_i} $(j_i \in \{1,2,\ldots N\}, N$ は運転手の人数) は山ダイヤ y_i を割り当てる運転手を表す。

ところがこの表現法では、$N < L$ であるから $d_{j_i} = d_{j_{i'}}$ となる i, i' が存在する。2 章で定義した組合せ最適化問題では、解は互いに異なるアルファベットの並びであることが必要である。

そこでアルファベットとして、運転手 d_{j_i} の代わりに、山ダイヤ y_i を割り当てる直前に d_{j_i} に割り当てられた山ダイヤ y_k を用いる。d_{j_i} に割当済みの山ダイヤがない場合は、d_{j_i} を用いる。つまりアルファベット s_i を、

$$s_i = \begin{cases} y_i & (i \leq N \text{のとき}) \\ d_{i-N} & (i > N \text{のとき}) \end{cases}$$

と定義し、解 σ_t (仕業ダイヤ) を

$$\sigma_t = s_{k_1} s_{k_2} \cdots s_{k_M} \quad (M = L + N)$$

と表現すれば、2 章で定義したヒューリスティック探索と同じになる。

そこで 3 章で提案した方法で、ヒューリスティック探索に GA を組み込んだ。すなわち、選択枝集合を十分大きくして、制約条件を満たす解を探索範囲に含まれるようにした上で、GA でヒューリスティック探索を最適化する。本手法により、すべての制約条件を満たすことが可能になり、仕業ダイヤ作成専門家が妥当と判断するレベルの仕業ダイヤを生成することが可能となった。

5 おわりに

ヒューリスティック探索を GA で最適化する手法を提案した。この手法を用いることにより、ヒューリスティックによる探索が既知である問題に、容易に GA を適用することができる。この結果、ヒューリスティックのチューニングが不要で開発工数が少ない、問題の諸条件が変化しても柔軟に適応可能、探索が大域的である、などの GA の長所が得られる。

この提案手法を、バス仕業ダイヤ作成へ適用した。ヒューリスティック探索のみによるバス仕業ダイヤ作成プログラムは、開発工数の都合上チューニングができなかったのであるが、GA を組み込むことにより実行可能解の生成が可能になった。ヒューリスティック探索の部分はバス会社に共通した基本的なロジックのみで構成されているのでステップ数の増大を避けることができ、保守性が向上した。また、特定のバス会社に依存した部分を持たないため、続いて受注した別のバス会社二社へ対応が容易であった。

スケジューリング、レイアウト問題など、知識工学が応用されている分野は数多い。これらの分野に GA が応用できれば、詳細にわたる知識獲得が不要となる。本報告で提案した手法を用いることにより容易に GA の応用分野を拡大することが出来るものと期待される。

参考文献

[1] 仙石, 吉原: 遺伝的アルゴリズムによる *TSP* の高速解法, 情報処理学会第 46 回全国大会 8D-4 (1993)

[2] 仙石, 吉原: 遺伝的アルゴリズムの最適解探索能力に関する評価 — *GA* と *SA* の比較 —, 情報処理学会第 47 回全国大会 (1993)

[3] Holland, J. H.: *Adaptation in Natural and Artificial Systems,* Univ. Michigan Press (1975)

[4] Goldberg, D. E.: *Genetic Algorithms in Search, Optimization, and Machine Learning,* Addison Welsley (1989)