動的重み付き最大制約充足問題に基づく ナーススケジューリングシステム

– 暫定制約の導入に基づく解の安定性の実現 –

Nurse Scheduling System based on Dynamic Weighted Maximal Constraint Satisfaction Problem

– Solution Stability based on the Provisional Constraint –

服部 宏充

Hirimitsu Hattori

日本学術振興会/名古屋工業大学大学院工学研究科情報工学専攻

Japan Society for the Promotion of Science/Department of Computer Science and Engineering, Graduate School of Engineering, Nagoya Institute of Technology

hatto@ics.nitech.ac.jp, <http://www-toralab.ics.nitech.ac.jp/~hatto/>

磯村 厚誌

Atsushi Isomura

東レ株式会社

Toray Industries, Inc.

Atsushi_Isomura@nts.toray.co.jp

伊藤 孝行

Takayuki Ito

名古屋工業大学大学院工学研究科情報工学専攻

Department of Computer Science and Engineering, Graduate School of Engineering, Nagoya Institute of Technology

itota@ics.nitech.ac.jp, <http://www-toralab.ics.nitech.ac.jp/~itota/>

大園 忠親

Tadachika Ozono

(同 上)

ozono@ics.nitech.ac.jp, <http://www-toralab.ics.nitech.ac.jp/~ozono/>

新谷 虎松

Toramatsu Shintani

(同 上)

tora@ics.nitech.ac.jp, <http://www-toralab.ics.nitech.ac.jp/~tora/>

keywords: dynamic weighted maximal CSP, nurse scheduling, stability

Summary

Scheduling has been an important research field in Artificial Intelligence. Because typical scheduling problems could be modeled as a Constraint Satisfaction Problem (CSP), several constraint satisfaction techniques have been proposed. In order to handle the different levels of importance of the constraints, solving a problem as a Weighted Maximal Constraint Satisfaction Problem (W-MaxCSP) is an promising approach. However, there exists the case where unexpected events are added and some sudden changes are required, i.e., the case with dynamic changes in scheduling problems. In this paper, we describe such dynamic scheduling problem as a Dynamic Weighted Maximal Constraint Satisfaction Problem (DW-MaxCSP) in which constraints would changes dynamically. Generally, it is undesirable to determine vastly modified schedule even if re-scheduling is needed. A new schedule should be close to the current one as much as possible. In order to obtain stable solutions, we propose the methodology to maintain portions of the current schedule using the provisional soft constraints, which explicitly penalize the changes from the current schedule. We have experimentally confirmed the efficacy of re-scheduling based on our method with provisional constraints. In this paper, we construct the nurse scheduling system for applying the proposed scheduling method.

1. は じ め に

スケジューリング問題の一つであるナーススケジューリングは、解決が困難なタスクとして知られている。なぜなら、ナーススケジューリングでは、重要度の異なる様々な条件を考慮に入れる必要があるためである。例えば、各看護師からの希望だけでなく、法的な規制や、人員配置数に関する規則などを考慮しなければならない [池

上 96]。ナーススケジューリングに対しては、制約充足問題 (Constraint Satisfaction Problem: CSP) [横尾 97] の枠組みに基づくアプローチが行われている [Abdennadher 99, Hofe 97]。例えば、文献 [Abdennadher 99] では、必ず満たさなければならない hard 制約と、必ずしも満たす必要のない soft 制約を、制約に付加する重みによって表現し、soft 制約の違反が最も少ない解の探索を行うための手法を提案している。また、具体的なシステ

図 1 スケジューリングシステムによる勤務表の例

ムとして INTERDIP を開発している。

ナーススケジューリングは、看護師から事前に報告される、勤務内容に関する要望に基づいて行われる。そのため、後日になって休暇申請が提出される等、スケジュールに対する要望の変化が起こり得る。その場合、再スケジューリングが必要となる。再スケジューリングを行う際の問題として、大幅なスケジュールの変更が行われ、その結果、看護師が混乱させられる点が挙げられる。そこで本論文では、看護師からの要望等によって変化する問題に対して、安定したスケジュールを決定可能な手法を提案し、大幅な変更を抑えたスケジュールを作成可能な、ナーススケジューリングシステムを実現する。本論文で提案するスケジューリング手法では、問題の変化を導入した CSP である動的 CSP [Dechter 88, Miguel 99] の枠組みに基づき、変化する問題を、複数の静的な CSP の列として表現する。ただし、ナーススケジューリングは、様々な条件(制約)を考慮する必要がある、一般に過制約な問題となる。そこで本手法では、各時点でのスケジューリング問題を、重み付き最大制約充足問題 (Weighted Maximal Constraint Satisfaction Problem: W-MaxCSP) [Larrosa 02, Lau 96] として定義し、動的に変化するナーススケジューリング問題を、動的重み付き最大制約充足問題 (Dynamic Weighted Maximal Constraint Satisfaction Problem: DW-MaxCSP) として新たに定式化する。本手法では、安定したスケジュールを提供するために、以前に決定されたスケジュールを維持するための暫定的な制約 (暫定制約) を導入し、再スケジューリングを行う。ここで用いる暫定制約は、DW-MaxCSP における各変数に対して、以前に割り当てられた値をできる限り維持することを明示した制約である。本論文では、暫定制約を導入することで、以前のスケジュールを大幅に変更せず、かつ解の質を落とすことなく、再スケジューリングが可能であることを示す。また、本手法が、本論文で実現するナーススケジューリングシステムにおいて、効果的に機能することを、実行例を用いて示す。

以下、本論文では、2 章で、本論文が対象とするナーススケジューリング問題について述べる。3 章で、動的

重み付き最大制約充足問題の定義を示し、動的に変化するスケジューリング問題の定式化を示す。次に、4 章では、動的重み付き最大制約充足問題において、解の安定性を実現するための、暫定制約の導入に基づく手法を提案し、具体的な再スケジューリングの手続きを示す。5 章で、提案手法に関する評価を示し、さらに本手法を導入したナーススケジューリングシステムについて、動作例を示す。そして、6 章で提案手法に関して考察を行い、7 章で関連研究について述べる。最後に、8 章でまとめと今後の課題を述べる。

2. ナーススケジューリング問題

看護師の勤務形態には 3 交代制と 2 交代制がある。3 交代制は、1 日を 3 つの勤務内容 (日勤 / 準夜勤 / 深夜勤) に分け、2 交代制では、2 つの勤務内容 (日勤 / 夜勤) に分ける。ナーススケジューリング問題とは、勤務形態に基づき、複数の看護師の勤務表を作成する問題である [池上 96]。本論文では、3 交代制に基づいてスケジューリングを行うものとする。3 交代制の下では、各看護師には、休日の場合と合わせて 4 通りのスケジュールのいずれかが割り当てられることになる。

図 1 に、本論文で試作するナーススケジューリングシステムが出力する勤務表の例を示す。図 1 では、各行が看護師の ID を示し、各列が勤務日を示している。本論文では、看護師の勤務形態を 3 交代制としているため、勤務表の各セルには、それぞれ $\square$: 日勤, $\circ$: 準夜勤, および $\triangle$: 深夜勤の値が割り当てられる。空白のセルは休日を示す。

ナーススケジューリングにおいては、様々な条件を考慮して勤務内容を割り当てる必要がある。考えられる条件として以下が挙げられる。

- 各勤務内容に従事する看護師数は、日付 (列) 毎に必要なだけ確保される。例えば、準夜勤には 3 人以上を確保し、深夜勤の人数は 4 人以下に抑える、といった条件が考えられる。
- 各看護師 (行) について、各勤務の回数はそれぞれ定められた範囲内でなければならない。例えば、休日は一週間に 1 人あたり 1 日 ~ 2 日とすること等が挙げられる。
- 看護師の勤務内容に対する希望を可能な限り満たす。すなわち、特定の日付の勤務内容の指定や、休日に関する要望が考慮されるべきである。
- 各看護師のスケジュールに、禁止される勤務パターンが含まれない。勤務パターンとは、看護師が数日間に渡り連続して就く勤務の並びである。例えば、文献 [池上 96] では、“深夜勤後の日勤”, および “深夜勤後の準夜勤” などが排除されるべきパターンとして挙げられている。
- 看護師の技術レベルを考慮し、新人のみでの勤務時

間帯などを禁止する．

以上のように，毎日の各看護師の予定は様々な条件により決定される．そのため，勤務表中のスケジュールを1つだけ変更した場合でも，これらの条件に関連する多くの変数に影響を及ぼす．筆者らは，スケジュールの変更による影響の大きさに着目し，動的重み付き最大制約充足問題に基づくアプローチにより，安定したスケジュールを提供可能な手法を提案する．

3. 動的重み付き最大制約充足問題に基づくナーススケジューリング

3.1 動的重み付き最大制約充足問題

本論文では，看護師からの新たな要望の提出が行われた場合の再スケジューリングを考慮する．新たな要望の提出によって変化する問題を表現するために，本論文では，動的 CSP [Dechter 88, Miguel 99] の枠組みを導入する．また，実世界におけるナーススケジューリングは一般に過制約であり，かつ各看護師の要望や，組織内での規則の重要性を明示的に区別する必要がある．そこで本論文では，制約に重みを付加し，違反する制約の重みの合計が最小化されるように解を求める動的 CSP，すなわち動的重み付き最大 CSP (DW-MaxCSP) として，再スケジューリングを考慮するナーススケジューリングを定式化する．

DW-MaxCSP は，静的な重み付き最大制約充足問題 (W-MaxCSP) の列として表現される．タイムステップ i における W-MaxCSP を WP_i とすると，DW-MaxCSP (DP) は以下のように表現できる．

$$DP = \{WP_0, WP_1, \dots, WP_i, \dots\}$$

ここで， WP_{i+1} は，直前の問題 WP_i に関して，制約の増減等が発生したことにより生成された問題となる．

個々の W-MaxCSP は， $WP_i = (X_i, D_i, C_i, S, \varphi)$ と表現される． (X_i, D_i, C_i) は従来の CSP であり，それぞれ変数，各変数の値域，および制約の集合を表す． $S = (E, \Sigma, \succ)$ は評価構造である． $\varphi: C_i \rightarrow E$ は評価関数であり，各制約に対して評価値を与える． E は可能な評価値の集合であり， $\succ$ は E の要素に全順序関係を定義する．また，最大/最小の評価値を，それぞれ $\top \in E$ ，および $\perp \in E$ とする． Σ は評価値を総計する．ここで， A を全変数への値の割り当てとすると，制約 c に関する割り当て A の評価値は以下のように定義される．

$$\varphi(A, c) = \begin{cases} \perp & \text{if } c \text{ is satisfied by } A \\ \varphi(c) & \text{otherwise} \end{cases}$$

そして，全ての制約を考慮した場合の割り当て A の評価値は，

$$\varphi(A) = \Sigma_{c \in C} \varphi(A, c).$$

となる．W-MaxCSP の解は，違反している制約の重みの総和 $\varphi(A)$ を最小化する，全変数への値の割り当てである．

動的 CSP において，変数に対して割り当てる値の変更を少なくするという条件を，解の安定性と呼ぶ [横尾 97]．解の安定性について，文献 [Verfaillie 94] では，前回の解と共通する割り当てを可能な限り多く維持する事とされている．また，文献 [Wallace 98] では，以後の問題の変化に対しても妥当性を維持できると予測できる解を安定した解としており，文献 [Verfaillie 94] で述べられているような安定性の考え方を “similarity” として，区別している．しかし，文献 [Wallace 98] における解の安定性は，similarity を包含する概念であり，解の有用性を維持しつつ問題の変化に対応する，という点で目的は同じである．本論文では，similarity のレベルで再スケジューリングを実現し，安定したスケジュールの作成を可能にする．

3.2 動的重み付き最大制約充足問題に基づく定式化

本節では，2章で述べたナーススケジューリング問題を DW-MaxCSP として定式化する．本論文が対象としている，動的な変化を伴うナーススケジューリングは，W-MaxCSP の列として定義でき，かつ個々の W-MaxCSP は，各タイムステップにおける，異なる変数とその値，および制約から成る．タイムステップ i のスケジューリング問題を表す W-MaxCSP WP_i を， $WP_i = (X_i, D_i, C_i, S, \varphi)$ と定義する．変数の集合 $X_i = \{x_{(1,1)}, x_{(1,2)}, \dots, x_{(s,t)}\}$ の各要素は，各日付における各看護師の勤務内容を表すとする．すなわち，各変数は，図1に示した勤務表の各セルに対応する．具体的には，要素 $x_{(s,t)} \in X_i$ は，看護師 s の，日付 t における勤務内容を表す． D_i は変数の値域集合を表すが，本論文におけるナーススケジューリングでは全変数で値域は共通とし， $d_{(s,t)} = \{0, 1, 2, 3\} \in D_i$ とする．ここでの値はそれぞれ， $0 =$ 休日， $1 =$ 日勤， $2 =$ 準夜勤，および $3 =$ 深夜勤である．また，評価構造 $S = (E, \Sigma, \succ)$ に関して， E は整数の集合であり， $\perp = 0$ ，かつ $\top = 9$ である．また， $\succ$ は通常的全順序関係とする．従って，全変数への完全な値の割り当てを A とすると， $\varphi(A)$ は違反制約の重みの和となる．

制約集合 C_i に含まれる制約の形式を以下のように定義する．

$$\lim(\min, \max, List, w)$$

ここで， $\min$ および $\max$ は，タイムステップ i において，全変数に割り当てられた値の集合 A_i と $List$ を比較した場合に，一致する要素数の下限 $\min$ と上限 $\max$ を表す． $List$ は，制約が要求する任意の変数の値の集合である． w は制約の重みであり， $0 \sim 9$ の整数値を取る．制約は， A_i と $List$ の間で一致する要素数 n が， $\min \leq n \leq \max$ である場合にのみ充足される．例えば，ある看護師 s が，

スケジューリングの対象期間中に、休日を 1 回以上、かつ 3 回以下得られることを希望する場合の制約は以下のように記述できる。

$$\lim(1, 3, \{x_{(s,1)} = 0, x_{(s,2)} = 0, \dots\}, 5)$$

つまり、この制約は、0 の値を割り当てられた変数の数が 1 以上 3 以下である場合に充足されることを意味している。また、この制約が違反された場合は 5 のコストが加算されることになる。本形式により、本論文におけるナーススケジューリングで扱う制約の表現が可能になる。変数に割り当てられた値が制約を充足するかどうかは、 A_i と $List$ のパターンマッチングに基づいて判定される。禁止される勤務パターンを制約として表現する場合、 $List$ に禁止される勤務パターンをそのまま記述すれば良い。例えば、“深夜勤務後の日勤”に関する制約の場合、 s および t が、日付を表す数値であるとして、 $List = \{x_{(s,t)} = 3, x_{(s,t+1)} = 1\}$ のように記述すれば良い。

最後に、本論文でのナーススケジューリングにおける問題の変化を以下に定める。DW-MaxCSP における各 W-MaxCSP 間の差異は、 $WP_i = (X_i, D_i, C_i, S, \varphi)$ の定義から、明らかに、変数、値域、および制約の変化と定義できる。上述の通り、一般のナーススケジューリングでは、値域の変更は考えられない。また、本論文では、スケジューリングの期間は変化せず、看護師の数の増減は無いと仮定する。そのため、本論文における、ナーススケジューリングの DW-MaxCSP としての定式化において、各 WP_i に発生し得る変化は、制約の追加 / 削除によってのみ発生することになる。従って、本節で述べた定式化では、看護師から新たな要望が提出された場合の制約の追加、および以前に提出した希望の修正による、制約の内容の変更に対応できる。制約の内容の変更は、以前提出された要望を表す制約を一度削除し、内容の変更を行った後に追加することで実現できる。

4. 動的重み付き最大制約充足問題に基づく再スケジューリング

4.1 暫定制約に基づく解の安定性の実現

前章までの定式化により、W-MaxCSP の列 $\{WP_0, WP_1, \dots, WP_i, \dots\}$ から成る DW-MaxCSP として、再スケジューリングを考慮するナーススケジューリングを表現できる。各タイムステップにおけるスケジューリング問題 WP_i は、通常の W-MaxCSP として解くことができる。各々の W-MaxCSP は、違反した制約の重みの合計が最小化されるように求解が試みられる。ここで、得られた解において、重みの低い制約をいくつか違反することで、直前に得られた解、すなわち WP_{i-1} の解からの変更が少なくなる可能性がある。例えば、看護師 s の日付 t の勤務内容を休日にするための、比較的弱い制約 $\lim(1, 1, \{x_{(s,t)} = 0\}, 1)$ が存在し、この制約を満たすた

めに、他の多数の変数の値が変更される場合が考えられる。制約 $\lim(1, 1, \{x_{(s,t)} = 0\}, 1)$ を満たす場合と満たさない場合とで、値が変更される変数の数が大幅に異なるならば、この制約に違反して、より変化の小さい解（すなわち安定した解）を得ることは、解の安定性に関して有効である。また、通常、スケジューリング問題 WP_{i-1} と WP_i は、解の計算に関して無関係であり、互いに影響を受けない。つまり、各問題は、個々に独立して求解が試みられる。従って、 WP_{i-1} と WP_i のように、前後の関係にある問題でも、得られるスケジュールは、互いに直接的に影響することなく、逆に大きく異なる場合も起こり得る。DW-MaxCSP における解の安定は、W-MaxCSP の列 $\{WP_0, WP_1, \dots, WP_i, \dots\}$ の個々の問題を逐次的に解く過程で、解が大幅に変化しないことで実現される。従って、直前に得られた解を何らかの方法で利用し、異なる W-MaxCSP 間に依存関係を発生させ、安定した解を得る方法が必要と考えられる。

以上の考察から、本論文では、各変数に対して、直前の問題の解において割り当てられた値と同じ値を持つ、という暫定制約（provisional constraint）を導入する手法を提案する。より具体的には、直前に得られた解自体を得るための重み付きの単項制約を付加する。たとえば、直前の問題において変数 $x_{(s,t)}$ に割り当てられた値を $v_{(s,t)}$ とすると、以下の暫定制約が追加される：

$$\lim(1, 1, \{x_{(s,t)} = v_{(s,t)}\}, w)$$

ここで、 w は、事前に定められた暫定制約に対する重みである。暫定制約の導入により、解の変更に対するペナルティを明示的に与えることができる。そのため、大幅な変更を要する重要度の低い制約に違反し、その代わりに暫定制約を充足する、といった解が計算可能である。その場合、変数の値の変更が少ない、安定した解を得ることができる。本手法では、直前に得られた解だけでなく、それ以前に得られた解を対象にして、近似した解を得ることを試みる。つまり、再スケジューリングが行われるたびに暫定制約を生成 / 追加し、各々の問題の解に近似した解が得られるようにする。すなわち、過去の再スケジューリングにおいて、ある看護師にとって望ましい勤務内容が変更された場合でも、変更前の値は暫定制約として保存され、勤務内容の変更を負担した事実が残される。そのため、以後の再スケジューリングにおいて、再び望ましい勤務内容が割り当てられる可能性を残すことができる。

4.2 再スケジューリングのプロセス

本節では、4.1 節で示した暫定制約の導入に基づくスケジューリングプロセスを示す。ここでは、看護師からの要望に基づく制約の集合 C_{new} 、および C_{rev} の追加により、スケジューリング問題 WP_i が、新たな問題 WP_{i+1} へと変化しているケースを考える。ここで、 C_{new} は、問

題 WP_i の解に対する新たな要望に基づく制約の集合であり, C_{rev} は, 問題 WP_i において提出した, 勤務内容に関する要望の変更に基づく制約の集合である. 再スケジューリングのプロセスは, 以下の Step 1 ~ Step 4 の 4 つのプロセスから成る.

Step 1: 看護師の要望から生成された, 再スケジューリングのきっかけになる制約の集合 C_{new} , および C_{rev} に基づき, スケジューリング問題 WP_i から, 新たな問題 WP_{i+1} を生成する.

Step 2: 全ての変数に関して, 問題 WP_i を解いて得られたスケジュールと同一の値を取るための, 複数の暫定制約の集合 C_{prov}^i を生成し, それまでのスケジューリングの過程で得られた暫定制約の集合 C_{prov} に追加する. すなわち,

$$C_{prov} = \bigcup_{j=0}^i C_{prov}^j$$

(ただし, $\forall j, c \in C_{prov}^j, c \notin (C_{prov} \setminus C_{prov}^j)$)

である.

Step 3: 暫定制約の集合 C_{prov} を WP_{i+1} に追加し, 仮の問題 WP'_{i+1} を生成する.

Step 4: Step 3 で生成された問題 WP'_{i+1} に対して, 山登り型の探索アルゴリズムを適用し, スケジュールの決定を行う. ここで, Step 2 における暫定制約の追加によって解の安定性は保証されるため, 新たな W-MaxCSP である WP'_{i+1} を任意の解法で解くことで, 可能な限り変更の割合を抑えたスケジュールを決定できる.

Step 5: 解の安定性のために追加された C_{prov} を取り除き, 問題を WP'_{i+1} から WP_{i+1} へ戻す. また, WP'_{i+1} を解いた際に充足された暫定制約を C_{prov} から取り除く. これは, 新たな問題 WP_{i+2} を解く必要が生じた場合に, 同一の暫定制約の重複を防ぐためである.

再スケジューリングアルゴリズムの詳細を図 2 に示す. ここでは, タイムステップ i におけるスケジューリング問題 WP_i が変化し, 新たな問題 WP_{i+1} が生成された場合のスケジューリングのアルゴリズムを示している. 1 行目では, 再スケジューリングを行う関数 `re_schedule` への入力として, タイムステップ i におけるナーススケジューリング問題 WP_i , WP_i における全変数への値の割り当て $\mathcal{A}_i$, 新たな要望に基づく制約の集合 C_{new} , 勤務内容に関する要望の変更に基づく制約の集合 C_{rev} , および暫定制約の集合 C_{prov} の 5 つが与えられている. 6 行目では, 4 行目で取り出された, スケジューリング問題 WP_i における勤務内容に関する希望を変更した看護師の ID に基づき, 変更前の希望を表す制約を WP_i における制約集合 C_i から取り除く. 8 行目から 11 行目にかけて, 問題 WP_i を解いて得られたスケジュールにおいて, 各変数に割り当てられた値 $v_{(s,t)}$ を維持するため

$WP_i = (X_i, D_i, C_i, S, \varphi)$: W-MaxCSP at time step i

$\mathcal{A}_i$: an assignment for WP_i

C_{new} : a set of constraints which is added to WP_i

C_{rev} : a set of revised constraints

C_{prov} : a set of provisional constraints

w : the weight of provisional constraint

```

1: re_scheduling( $WP_i, \mathcal{A}_i, C_{new}, C_{rev}, C_{prov}$ )
2:  $ID_{new} \leftarrow$  ID of nurse associated with  $c \in C_{new}$ 
3:  $DT_{new} \leftarrow$  date associated with  $c \in C_{new}$ 
4:  $ID_{rev} \leftarrow$  ID of nurse associated with  $c \in C_{rev}$ 
5:  $DT_{rev} \leftarrow$  date associated with  $c \in C_{rev}$ 
6:  $C_i \leftarrow C_i - \{c\}$  ( $c$  is a constraint on the desire associated
   with  $s \in ID_{rev}$ )
7:  $C_{i+1} \leftarrow C_i \cup C_{new} \cup C_{rev}$ 
8: for each  $x_j \in X_i$ 
9:   if  $s \notin ID_{new} \vee t \notin DT_{new} \vee s \notin ID_{rev} \vee t \notin DT_{rev}$  then
10:     $C_{prov} \leftarrow C_{prov} \cup \text{lim}(1, 1, \{x_j = v_j\}, w)$ 
11:   end for
12:  $C'_{i+1} \leftarrow C_{i+1} \cup C_{prov}$ 
13:  $\mathcal{A}_{i+1} \leftarrow \text{hill\_climbing}(X_i, D_i, C'_{i+1}, S, \varphi)$ 
14: for each  $c \in C_{prov}$ 
15:   if ( $c$  is satisfied in  $\mathcal{A}_{i+1}$ ) then
16:     $C_{prov} \leftarrow C_{prov} - \{c\}$ 
17:   end for
18:  $WP_{i+1} \leftarrow (X_i, D_i, C_{i+1}, S, \varphi)$ 
19: return  $WP_{i+1}, \mathcal{A}_{i+1}$  and  $C_{prov}$ 

```

図 2 再スケジューリングのアルゴリズム

の暫定制約を新たに生成し, C_{prov} へ格納している. ここで, C_{new} , および C_{rev} 中の制約によって変更が試みられている変数に関しては, 暫定制約は生成しない. 図 2 に示したアルゴリズムでは, 暫定制約の重み w は全て同一とし, 事前に定義されているが, 必要に応じて異なる重みを与えることも可能である. 13 行目では, 山登り法に基づいて W-MaxCSP を解く関数 `hill_climbing` により, タイムステップ $i+1$ における各変数への値の割り当てを決定する. ここで, 関数 `hill_climbing` の引数 $(X_i, D_i, C'_{i+1}, S, \varphi)$ は, WP'_{i+1} を表している. C'_{i+1} は, 12 行目で生成された, 通常の制約の集合 C_{i+1} と暫定制約の集合 C_{prov} から成る制約の集合である. 14 行目から 17 行目では, 得られたスケジュール $\mathcal{A}_{i+1}$ において充足された暫定制約のみを, C_{prov} から取り除いている. 最後に 19 行目で, 関数 `re_scheduling` の出力として, 新たに生成された問題 WP_{i+1} , 全変数への値の割り当て $\mathcal{A}_{i+1}$, および暫定制約の集合 C_{prov} を返す.

図 3 に, 関数 `re_scheduling` から呼び出している, W-MaxCSP の解の探索アルゴリズムを示す. 本アルゴリズムでは, ヒューリスティックを用いて解の改善を行う山登り型の探索アルゴリズムに基づいて解を計算する. 関数 `hill_climbing` では, 変数の集合 $X = x_0, x_1, \dots$ と, 各変数に割り当てる値の集合 $V = v_0, v_1, \dots$ から成る解

$WP_i = (X_i, D_i, C_i, S, \varphi)$: W-MaxCSP at time step i

```

1: hillclimbing ( $WP_i$ )
2:   $s(X, V) \leftarrow \text{set\_initial}(WP_i)$ 
3:   $V_{best} \leftarrow V$ 
4:  repeat
5:    if  $\text{eval}(s(X, V)) = 0$  then
6:      return  $s(X, V)$ 
7:    for  $i = 0$  to  $|X|$ 
8:       $v'_i \leftarrow \arg \max_{v \in d_{x_i}} \text{eval}(s(X, V \cup \{v\} - \{v_i\}))$ 
9:       $V' \leftarrow V \cup v'_i - \{v_i\}$ 
10:     if  $\text{eval}(s(X, V')) \leq \text{eval}(s(X, V_{best}))$  then
11:        $V_{best} \leftarrow V'$ 
12:   end for
13:   if  $\text{eval}(s(X, V_{best})) < \text{eval}(s(X, V))$  then
14:      $V \leftarrow V_{best}$ 
15:   else
16:      $V \leftarrow \text{heuristic\_assignment}(C_i, V)$ 
17: until cutoff steps
18: return  $V$ 

1: heuristic_assignment( $C, V$ )
2:   $c \leftarrow$  a constraint which is randomly selected from  $C$ 
3:  if there are some unsatisfied variable in List of  $c$  then
4:     $x_c \leftarrow$  an unsatisfied variable selected from List of  $c$ 
5:     $v_c \leftarrow$  the required value of  $x_c$  in List of  $c$ 
6:     $v'_c \leftarrow$  the value of  $x_c$  in  $V$ 
7:     $V' \leftarrow V \cup \{v_c\} - \{v'_c\}$ 
8:  else
9:    heuristic_assignment( $C, V$ )
10: return  $V'$ 

```

図 3 ヒューリスティックスに基づく山登り型探索アルゴリズム

$s(X, V)$ に対して、関数 `set_initial` で初期の暫定解を設定する (2 行目)。そして、違反制約が皆無の解を発見するか、事前に設定したループ数 (cutoff steps) に達するまで、違反制約が少なくなる方へ解を改善する。ここで、7 行目から 12 行目における処理によって解が改善されなかった場合、関数 `heuristic_assignment` によって、変数への値の割り当てをヒューリスティックスに更新し、解の改善を促進する。

関数 `heuristic_assignment` では、制約の集合 C からランダムに制約 c を取り出し (2 行目)、制約 c の *List* における要求が満たされているか確認する (3 行目)。ここで、 V において割り当てられている値が、*List* で求めている値と異なる変数 x_c が存在する場合、その変数の値を、*List* で求めている値に更新する (4 行目 ~ 7 行目)。例えば、 V において $x_{(s,t)} = 1$ である場合に、制約 c の *List* が $\{x_{(s,t)} = 2\}$ であるならば、変数 $x_{(s,t)}$ の値は 2 に更新される。以上の処理により、制約中で求められている値を、変数に積極的に割り当てることで、制約の充足を促進できる。また、*List* の要求が全て満たされてい

る場合、再帰的に関数 `heuristic_assignment` を呼び出す (9 行目)。

本節で述べたスケジューリング手法において、ヒューリスティックスを用いて解を改善する山登り型の探索アルゴリズムを利用した理由に関して、以下に述べる。本論文では、実際の病院において発生し得る、突発的なスケジュールの変更に対応可能なナーススケジューリングシステムの実現を目的としている。本システムでは、スケジューリング問題の変化によって、数回のスケジューリング処理が行われる。従って、実用性の点から、スケジューリング処理にかかる時間は、より短い方が良い。山登り型の探索は単純なアルゴリズムであり、高速に解を決定できる。その一方で、山登り型の探索では解の最適性が保証されない。しかし、実在の病院では、手作業によって勤務表が作成されており、ここでは明らかに、解の最適性は保証されていない。従って、現実的には、ナーススケジューリングにおいて厳密な最適解を得ることは必須の条件ではないと考察できる。

以上により、本論文では、より高速な処理が可能である点から山登り型の探索アルゴリズムを採用し、かつ制約の充足を促進するためのヒューリスティックスを導入した。ヒューリスティックスの導入により、品質の良い解を効率的に得ることを試みている。ここでは、ヒューリスティックスの適用対象を、より重要度の高い制約に絞り込むことにより、解の品質の向上が期待できる。より具体的には、図 3 の関数 `heuristic_assignment` の 2 行目において、高い重要度を持つ制約の集合から制約 c を取り出すようにすれば良い。より高い品質の解を得るための方法として、例えば、最適解を計算可能な分岐限定法の適用が考えられる。しかし、上記の考察から、実用上、多くの処理時間をかけて最適解を得る必要性は低い。処理時間と解の品質はトレードオフの関係にあり、スケジュールの探索アルゴリズムの決定は容易ではない。適切な探索アルゴリズムの開発は今後の課題である。

5. 評価

5.1 暫定制約に基づく再スケジューリング手法の評価

本論文で提案したスケジューリング手法の有用性を示すため、次の条件の下で実験を行なった。本実験では、スケジューリングの対象期間を 2 週間、看護師の人数を 15 人とし、合計 210 個の変数に対して、日勤 / 準夜勤 / 深夜勤 / 休日のいずれかの値を割り当てる。スケジュールは、以下の条件を考慮して決定される。

(i) 必ず満たすべき条件

- 1 日の各勤務内容に必要な人数を満たすこと (日勤: 4~6 人, 準夜勤: 3~5 人, 深夜勤: 3~5 人)
- “深夜勤後の日勤”, “深夜勤後の準夜勤”, “準夜勤後の日勤”, および “3 連続夜勤” を、禁止すべき勤務パターンとして避けること

図 4 変化率の比較

図 5 評価値の比較

(ii) 満たすことが望ましい条件

- 看護師の各勤務回数，および休日の日数に関する条件を満たすこと（日勤／準夜勤：2日以上，深夜勤：4日以内，休日：3～5日）
- “4連続準夜勤”，“5連続日勤”，および“前後が休日となる勤務”を，無いことが望ましい勤務パターンとして避けること

(i) の条件に関する制約の重みを 9，(ii) の条件に関する制約の重みを 4 と設定する．また，各看護師に関して，2週間分の勤務内容に関する希望を表す制約を生成する．本制約に関する重みは，[1,9] で一様に分布していると仮定し，各制約に対して，1～9 のいずれかの値をランダムに設定する．

本実験では，まず，以上の設定に基づき，図 3 のアルゴリズムを用いて，勤務表を作成する．ここで，関数 `set_initial` で与える初期の暫定解は，各看護師が希望する勤務内容とする．次に，ランダムに選択した勤務表中のセルに関して，勤務内容の変更を要求する制約を新たに生成し，再スケジューリングを行う．変更要求を極力受理するために，生成する制約の重みは 9 に設定する．提案手法との比較対象として，再スケジューリングであることを考慮せず，新たなスケジューリング問題として

単純に計算を行う手法（単純再計算），および前回の問題を解いて得られた解を初期値として利用する，初期値利用法 [Minton 92, Selman 92, Verfaillie 94]（初期値利用）を用いる．提案手法は以下の条件で実行する：

- 違反制約の重みの総和が 0 にならない場合に，探索処理を繰り返す回数の上限は 800 回に設定する
- 探索の際の初期の暫定解は，各看護師が希望する勤務内容とし，単純再計算の場合と提案手法を用いる場合で初期値を同一にする
- 関数 `heuristic_assignment` において変数値の更新を行う制約を，各勤務内容に必要な人数を表す制約とする
- 追加する暫定制約の重みは 2 に設定する

変更を要求する制約数を 3 個から 30 個まで変化させた場合の，変化率と評価値の推移を図 4，および図 5 に示す．変化率とは，再スケジューリングによって値が変更された変数の数の，全変数に対する割合である．図 4，および図 5 の結果は，制約数が異なる各々のケースに関して，30 回試行した結果の平均値である．図 4 のグラフの横軸は，変更を要求する制約の数を表し，縦軸は変化率を表す．図 5 のグラフの横軸も同様に変更を要求する制約の数を表し，縦軸は評価値を表す．ただし，図 5 に示す提案手法の評価値は，暫定制約を除いて計算された値である．図 4 に示す通り，解の安定性が全く考慮されない単純再計算では，常に 6 割以上の変数の値が変更される．初期値利用では，単純再計算よりも変化率は 2 割程度低い，約半数の変数の値が変更される．つまり，これら 2 つの手法では，制約数に関わらず，要求の対象では無い多くの変数が値を変更されてしまう．しかし，提案手法では，制約数の増加に伴って変化率は増加するが，最も制約数が少ない場合で約 1 割，制約数が最も多い場合で約 3 割と，変化率を低く抑えることができ，他の 2 手法と比較して，高い解の安定性が得られている．また，変化率の増加は線形であり，制約数の増加に伴って，急激に値が悪化することも無い．

一方，図 5 に示す通り，提案手法から得られる評価値は，初期値利用の場合よりも良く，単純再計算よりも悪いが，大きな差は見られない．すなわち，提案手法は，前回のスケジュールからの変更の度合いを抑えつつ，良品質の解を得ることを可能にしている．ここで，制約数が多い場合に，初期値利用の評価値が悪化しているのは，多くの変数に対して値の変更が要求されたため，前回得られた解からの反復改善が有効でなくなったためと考えられる．また，単純再計算の場合の評価値が提案手法より良好なのは，提案手法では，解の安定性を得るための暫定制約が存在するため，より評価値の高い解を得ることができないためと考えられる．

次に，表 1 に単純再計算，初期値利用，および提案手法の処理時間を示す．表 1 では，30 個の変数に対して値の変更が要求されるケースに関して，30 回試行した結果

表 1 平均処理時間の比較

計算手法	単純再計算/初期値利用	提案手法
処理時間	487 sec.	693 sec.

の平均処理時間を示している．単純再計算と初期値利用では，処理すべき制約数が同じになるため，処理時間はほぼ同一になる．そのため，表中では処理時間を区別していない．また，表 1 では，個々の問題に対して，10 回目の再スケジューリングを行う際の処理時間を示す．単純再計算/初期値利用では，再スケジューリングを行った回数によって処理時間は変化しない．しかし，4.2 節で述べた通り，提案手法では，再スケジューリングを行うごとに暫定制約の数が増加するため，処理時間は大きくなる．本実験結果は，PowerPC G5 1.6GHz を搭載した計算機上で計測した．表 1 に示す結果より，提案手法が他の手法よりも余分に必要とする処理時間は，スケジューリングを 10 回繰り返した場合でも高々 200 sec. 程度であり，実用上の問題は無い．また，図 4 と図 5 に示した解の安定性に関する性能は，提案手法における処理時間の増加を十分に補うものであると言える．

5.2 システム実行例

本節では，本論文で提案したスケジューリング手法に基づくナーススケジューリングシステムの実行例を示す．図 6，および図 7 は，本システムで実際に作成された，再スケジューリング前/再スケジューリング後の勤務表を示している．図 6 は，スケジューリングを行う前の勤務表である．表中のセルには，この時点でのスケジュールにおける，看護師の勤務内容が表示されている．ここで，表中に 4 つ存在する色が反転したセルは，勤務内容の変更が要求されている事を示している．新たに要求されている勤務内容は，“[]”内に表示されている．例えば，“nurse2”は，2 日の勤務内容を準夜勤から夜勤に変更することを要求しており，“nurse10”は，準夜勤から休日に変更することを要求している．図 7 は，本システムによって再スケジューリングを行った結果を示している．図 7 において，円で囲まれたセルは，再スケジューリングによって勤務内容が変更された事を表している．また，変更を要求されたセルに関して，勤務内容の変更に成功したセルは図 6 と同様，セルの色を反転して表示されている．図 7 の例では，全ての変更要求の反映に成功していることが分かる．

本節における実行例は，5.1 節の実験と同様，看護師 15 人に関する 2 週間分の勤務表であるため，合計 210 個の変数で構成されている．本実行例では，4 つの変数に関して値の変更が要求されている．変更要求を満たすために，再スケジューリングによって，他の 206 個の変数においても値の変更が発生し得る．ここで，もしも，再スケジューリングによって，変更を要求していない看護師の勤務内容に大きな影響を与えてしまうならば，安

図 6 再スケジューリングの例：変更前

図 7 再スケジューリングの例：変更後

定したスケジュールが得られるとしても，やはり看護師を混乱させてしまう．この点を評価するため，図 8 に，再スケジューリングによって勤務内容が変更された変数の割合（変更率）を，各看護師ごとに表示した．ここでは，30 個の異なる問題を対象として，図 6 と同様のセル（変数）に対して値の変更を要求した．変更率は，値の変更を要求した変数を除いて計算した値である．グラフの横軸の数値は各看護師の ID を表し，縦軸は変更率を表す．本評価では，“nurse2”，“nurse6”，“nurse10”，および“nurse14”が勤務内容の変更を要求しており，図 8 では，これらの看護師のグラフを強調して表示している．図 8 に示す通り，再スケジューリングの結果，変更を要求した 4 人の看護師に対しては，他の看護師と比較して，より多くの勤務内容の変更が行われる．すなわち，変更を要求した変数以外での値の変更が，高い頻度で行われている．この理由について，以下に考察する．本論文では，制約の種類として，(i) 複数の看護師の勤務内容を関連付ける制約，および (ii) 個々の看護師の勤務内容に関する制約，の 2 つを考慮している．本論文の問題設定では，(i) の種類の制約は，“一日の各勤務内容について必要な人数を満たす”ための制約のみが用いられている．一方，(ii) の種類の制約は，看護師の各勤務内容の回数や勤務パターンに関する，複数の制約が用いられており，(i) の種類の制約よりも数が多い．そのため，ある看護師 s が勤務内容の変更を要求した場合，看護師 s 自身

図 8 各看護師の勤務内容の変更率

に関する，より多くの制約が影響受けることになる．その結果，看護師 s に関連する変数，すなわち勤務表における看護師 s の行において，勤務内容の変更が発生しやすくなったと考えられる．図 7 の実行例においても，変更を要求した看護師（“nurse2”，“nurse6”，“nurse10”，および “nurse14”）は，他の看護師よりも勤務内容の変更が多く行われている．従って，本論文で提案したスケジューリング手法の導入により，本ナーススケジューリングシステムでは，安定したスケジュールを得られるだけでなく，勤務内容の変更を要求していない看護師に対して大きな影響を与えないため，看護師の混乱を回避できると言える．また，本システムにおいて勤務内容の変更を要求する看護師は，勤務内容の変更を強いられる可能性が高くなることを認識する必要がある．

6. 考 察

5.1 節において，図 4，および図 5 に結果を示した実験は，暫定制約の重みを 2 として行った．暫定制約の重みが小さ過ぎる場合，解の安定性を得ることが難しくなる．また，大き過ぎる場合，以前の解から変化せず，要求する変数の値の変更が困難となり，評価値が悪化する．このように，暫定制約の重みの決定にはトレードオフが存在し，適切な値の決定は困難な問題である．そこで本節では，暫定制約に設定する重みを変化させた場合の変化率，および評価値の変化に関して評価し，考察を行う．

図 9 に，暫定制約に設定する重みを変化させた場合の，変化率と評価値の推移を示す．ここでは，12 個の変数に対して値の変更を要求する問題において，暫定制約の重みを 1 から 7 まで変化させた場合の評価値と変化率を示す．図 9 の横軸は暫定制約の重みを表し，左側の縦軸は評価値，右側の縦軸は変化率を表す．ただし，評価値は，重みが 1 の場合の評価値に対する比率で表されている．

図 9 に示す通り，重みが 1 から 4 までの間では，評価値 / 変化率は緩やかに値が変化しているが，重みが 4 と

図 9 暫定制約の重みに関する評価値 / 変化率の推移

5 の場合の間で，評価値 / 変化率に大きな変化が見られる．評価値は約 2 倍悪化しており，変化率も約 1 割低下している．これは，本論文では，“満たすことが望ましい条件”を表す制約の重みを 4 に設定しているため，暫定制約の重みが 4 を越えたことで，それらの制約が充足されにくくなったためと考えられる．5.1 節における評価では，暫定制約の重みを 2 に設定しているが，図 9 に示す結果から，解の安定性を実現しつつ，良質な解を得るためには適切な設定であったことが分かる．このように，暫定制約の重みに設定する適切な値は，他の制約の重みの設定に依存する．本論文では，看護師の希望を表す制約の重みは，[1,9] で一様に分布すると仮定しているが，例えば，重みの分布が 4 以下の値に偏った場合，1 から 4 の間の評価値 / 変化率の推移は，図 9 よりも大きくなる．

以上の考察から，暫定制約の重みの設定は，他の制約の重みの設定と併せて慎重に行う必要がある．しかし，例えば，看護師の希望を表す制約の重みが，何らかの規範（*e.g.*, 勤続年数）に基づいて事前に決定されるならば，暫定制約の重みを適切な値に設定することは可能と考えられる．

7. 関 連 研 究

本論文では，動的 CSP の枠組みに基づくスケジューリング手法を提案し，特に，求解の過程での解の安定性に注目している．解の安定性に関する研究として，文献 [Miguel 03, Schiex 93, Verfaillie 94, Wallace 98] がある．

文献 [Verfaillie 94] では，変化する前の問題 P_i と，変化した後の問題 P_{i+1} 間でのみ，解の安定性を考慮している．つまり，個々の再スケジューリングでの変化を低く抑えることができれば，将来的にも安定した解を獲得し続けることができると仮定している．一方，3.1 節でも述べた通り，文献 [Wallace 98] における解の安定性の定義は，文献 [Verfaillie 94] とは異なり，以後の問題の変化に対しても妥当性を維持できると予想できる解を安定

した解としている。すなわち、以後の計算において、更に変更される可能性が小さい解ほど、安定性が高い解となる。安定性の高い解を得るために、文献 [Wallace 98] では、値の割り当てに対してペナルティを与える形式をとっている。問題が変化し、新たな制約違反が発生した場合、制約違反に関係する変数が割り当てられている値にペナルティを与える。次回以降の計算時には、ペナルティの大きい割り当てを回避する。以上のサイクルを繰り返すことで、将来的に、変更の割合が小さい解を得ることが可能としている。文献 [Wallace 98] では、問題の変化を 250 回繰り返し、良好な解が得られることを確認している。本論文で対象としているナーススケジューリングでは、100 回を越えるような多数回の問題の変化は発生しないと考えられる。文献 [Wallace 98] の手法では、初期の段階でのペナルティの値は、安定した解を得るためには適切では無いと言える。従って、本論文で対象としているナーススケジューリング問題には適していない。

また、文献 [Hentenryck 91, Kleer 89, Schiex 93] では、安定した解を計算可能な手法として、制約記録法と呼ばれる手法が用いられている。本手法では、以前の問題の解を求める過程で導かれた制約と、その制約が成立するための前提条件を記録しておく。問題が変化した場合、前提条件が無効となった制約が取り除かれる。しかし、本手法は、解の安定性よりも、求解の効率化に関しより注力した手法である。

文献 [Miguel 03] では、過制約かつ動的に変化する問題を表現するためにファジィ CSP と動的 CSP の枠組みを組み合わせ、fuzzy restriction/relaxation Dynamic Flexible CSP (fuzzy rrDFCSP) を提案している。文献 [Miguel 03] では、fuzzy rrDFCSP を用いることで、問題の変化を考慮しない手法と比較して、良好な結果が得られることを実験的に示している。しかし、本実験では 2 項制約のみを考慮しており、また変数の数も最大 40 個である。一方、本論文では、より複雑かつ大規模な問題を対象にして実験を行い、提案手法が、実用上、高い有用性を持つことを示している。

制約充足問題に基づいてナーススケジューリングにアプローチした研究として、文献 [Abdennadher 99]、および [Hofe 97] の研究がある。

文献 [Abdennadher 99] では、ナーススケジューリングを、不完全制約充足問題 (partial CSP) [Freuder 92] を拡張した、重み付き制約充足問題として定式化している。また、スケジューリング問題を解くために、論理プログラミングと制約解消の 2 つの宣言的パラダイムの利点を持った、制約論理プログラミング (CLP) に着目し、CLP の技術に基づくナーススケジューリングシステム INTERDIP を開発している。文献 [Abdennadher 99] では、解を計算するための探索アルゴリズムとして、独自に改良した分岐限定法を用いている。ここでは、探索空間を削減するために (1) 休日の割り当て (2) 深夜勤

の割り当て、および (3) 日勤と準夜勤の割り当て、という 3 つのフェーズに問題を分割している。このモデル化によって、各々のフェーズでは、勤務表の各セルに関して、二者択一という最小の決定をするだけで良く、探索空間の削減によるアルゴリズムの効率化を実現している。

文献 [Hofe 97] では、ナーススケジューリング問題を、600 個から 800 個の変数から構成される階層的制約充足問題 (Hierarchical Constraint Satisfaction Problem: HCSP) として定式化している。本文献では、制約の重要度を重み (weight) と優先度 (priority) という 2 つの実数値によって表現しているが、HCSP に基づく定式化によって、これら 2 つの数値の統合を実現している。ここでは、優先度の高い制約は、それぞれ、より優先度が低い全ての制約よりも重要であるとし、各優先度において、制約は重みによって順序付けられる。すなわち、優先度に基づいて制約の階層化を行い、重みに基づいて各階層における複数の制約間の順序付けを行っている。本定式化に基づいて、可能な限り優先度の高い制約を充足しつつ、制約の重みの和を最大化する解を見つけるために、文献 [Hofe 97] では、階層的制約論理プログラミング (Hierarchical Constraint Logic Programming: HCLP) を導入している。また、求解の方法に関して、探索の効率化のために、ヒューリスティックな探索アルゴリズムを導入している。文献 [Abdennadher 99] と [Hofe 97] は共に、ヒューリスティックの導入による探索時間の削減によって、実用性を向上させている。本論文の提案手法においても、ヒューリスティックの導入によって解の探索の効率化を行っている。本論文で導入したヒューリスティックは、探索の効率化だけでなく、解の品質の向上も期待できる。

8. ま と め

本論文では、再スケジューリングが可能なナーススケジューリングシステムの構築を試みた。再スケジューリングでは、以前に決定したスケジュールから大幅な変更が行われ、看護師にとって不都合が生じる、といった実用上の問題が存在する。そこで本論文では、まず、一般的に過制約となるナーススケジューリング問題を重み付き最大制約充足問題として表現した。そして、動的制約充足問題の枠組みに基づき、動的重み付き最大制約充足問題として、動的に変化するナーススケジューリング問題を定式化した。ここで、以前のスケジュールに近似したスケジュールを得るための解の安定性の実現手段として、本論文では、暫定制約を導入した。暫定制約の導入により、解の変更に関するペナルティを明示的に扱うことができ、スケジュールの変更に対して制約を設けることが可能となった。本論文では、暫定制約に基づくスケジューリング手法によって、良質であり、かつ大幅な変更の無い安定したスケジュールが得られることを、実験

によって示した。また、提案手法では、より多くの制約を扱うことになるが、十分に実用的な時間でスケジュールが得られることを示した。

本論文では、暫定制約に対して与える重みに関して、問題設定と併せて考慮する必要があることを考察した。そこで、本研究の今後の課題として、暫定制約の重みの適切な決定手法の開発が挙げられる。特に、本論文では、追加する全ての暫定制約の重みを同一としていたが、制約ごとに異なる重みを与える方が効果的である問題も存在すると考えられる。従って、個々の暫定制約に適切な重みを設定する手法について、研究を行う必要がある。

◇ 参 考 文 献 ◇

- [Abdennadher 99] Abdennadher, S., Schlenker, H.: Nurse Scheduling using Constraint Logic Programming, in Proc. of the 11th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI-99), pp. 838-843, 1999.
- [Dechter 88] Dechter, R., Dechter, A.: Belief Maintenance in Dynamic Constraint Networks, in Proc. of the 7th National Conference on Artificial Intelligence (AAAI-88), pp. 37-42, 1988.
- [Freuder 92] Freuder, E. C., and Wallace, R. J.: Partial Constraint Satisfaction, Artificial Intelligence Journal, Vol. 58, No. 1-3, pp. 21-70, 1992.
- [Hentenryck 91] Hentenryck, P. V., and Provost, T. L.: Incremental Search in Constraint Logic Programming, New Generation Computing, Vol. 9, pp. 257-275, 1991.
- [Hofe 97] Hofe, H. M.: ConPlan/SIEDAplan: Personnel Assignment as a Problem of Hierarchical Constraint Satisfaction, in Proc. of the 3rd International Conference on Practical Applications of Constraint Technologies (PACT-97), pp. 257-272, 1997.
- [池上 96] 池上敦子, 丹羽明, 大倉元宏: 我が国におけるナース・スケジューリング問題, オペレーションズ・リサーチ, Vol. 41, No. 8, pp. 436-442, 1996.
- [Kleer 89] de Kleer, J.: A Comparison of ATMS and CSP Techniques, in Proc. of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89), pp. 290-296, 1989.
- [Larrosa 02] Larrosa, J.: Node and Arc Consistency in Weighted CSP, in Proc. of the 18th National Conference on Artificial Intelligence (AAAI-02), pp. 48-53, 2002.
- [Lau 96] Lau, H.C.: A New Approach for Weighted Constraint Satisfaction: Theoretical and Computational Results, in Proc. of the 2nd International Conference on Principles and Practice of Constraint Programming (CP-96), pp. 323-337, 1996.
- [Miguel 03] Miguel, I., and Shen, Q.: Fuzzy τ DFCSP and Planning, Artificial Intelligence Journal, Vol. 148, No. 1-2, pp. 11-52, 2003.
- [Miguel 99] Miguel, I., Shen, Q.: Hard, Flexible and Dynamic Constraint Satisfaction, Knowledge Engineering Review, Vol. 14, No. 3, pp. 285-293, 1999.
- [Minton 92] Minton, S., Johnston, M., Philips, A., and Laird, P.: Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problem, Artificial Intelligence Journal, Vol. 58, No. 1-3, pp. 160-205, 1992.
- [Schiex 93] Schiex, T., and Verfaillie, G.: Nogood Recording for Static and Dynamic Constraint Satisfaction Problems, in Proc. of the 5th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-93), pp. 48-45, 1993.
- [Selman 92] Selman, B., Levesque, H., and Mitchell, D.: A New Method for Solving Hard Satisfiability Problems, in Proc. of the 18th National Conference on Artificial Intelligence (AAAI-92), pp. 440-446, 1992.
- [Verfaillie 94] Verfaillie, G., Schiex, T.: Solution Reuse in Dynamic Constraint Satisfaction Problems, in Proc. of the 20th National Conference on Artificial Intelligence (AAAI-94), pp. 307-312, 1994.
- [Wallace 98] Wallace, R.J., Freuder, E.C.: Stable Solutions for Dynamic Constraint Satisfaction Problems, in Proc. of the 4th International Conference on Principles and Practice of Constraint Programming, pp. 447-461, 1998.
- [横尾 97] 横尾真, 平山勝敏: CSP の新しい展開: 分散 / 動的 / 不完全 CSP, 人工知能学会誌, Vol. 12, No. 3, pp. 33-41, 1997.

〔担当委員: 武田英明〕

2004 年 3 月 22 日 受理

著 者 紹 介

服部 宏充(正会員)

2004 年名古屋工業大学大学院工学研究科電気情報工学専攻博士後期課程修了。2004 年から日本学術振興会 特別研究員 (PD)。2004 年 10 月より英国リバプール大学 Honorary Research Assistant。現在に至る。マルチエージェントシステム, 電子商取引支援, 意思決定支援, 特にマルチエージェントの協調・交渉メカニズムに興味を持つ。博士 (工学)。2003 年度情報処理学会全国大会 大会奨励賞受賞。AAAI, 人工知能学会, 電子情報通信学会, 情報処理学会, 日本ソフトウェア科学会各会員。

磯村 厚誌

2002 年名古屋工業大学工学部知能情報システム学科卒業。2004 年名古屋工業大学大学院工学研究科電気情報工学専攻博士前期課程修了。同年, 東レ (株) 入社。現在に至る。

伊藤 孝行(正会員)

2000 年名古屋工業大学大学院工学研究科電気情報工学専攻博士後期課程修了。1999 年から 2001 年にかけて日本学術振興会特別研究員 (DC2, PD)。2000 年から 2001 年にかけて南カリフォルニア大学情報科学研究所客員研究員。2001 年から 2003 年にかけて北陸先端科学技術大学院大学知識科学教育研究センター助教授。2003 年名古屋工業大学大学院工学研究科助教授。現在に至る。博士 (工学)。2003 年度情報処理学会全国大会奨励賞受賞。マルチエージェントシステム, オークション理論, 電子商取引支援に関心を持つ。AAAI, ACM, 人工知能学会, 電子情報通信学会, 情報処理学会, 計測自動制御学会, 日本ソフトウェア科学会各会員。

大園 忠親(正会員)

1995 年名古屋工業大学工学部知能情報システム学科卒業。2000 年同大学大学院工学研究科電気情報工学専攻博士後期課程修了。同年より同大学 工学部知能情報システム学会助手。2003 年から 2004 年にかけてマレイシア・マルチメディア大学客員研究員。現在に至る。博士 (工学)。情報エージェントの研究に従事。AAAI, ACM, 電子情報通信学会, 人工知能学会, 情報処理学会, 日本ソフトウェア科学会各会員。

新谷 虎松(正会員)

1982 年東京理科大学大学院修士課程修了。同年富士通 (株) 国際情報社会科学研究所入所。知識システム, 論理プログラミングなどの研究に従事。1993 年名古屋工業大学知能情報システム学科助教授。1999 年同大学同学科教授。1999 年から 2000 年にかけて米国カーネギーメロン大学ロボティクス研究所客員研究員。現在に至る。博士 (工学)。2003 年度情報処理学会全国大会優秀賞受賞。意思決定支援システム, マルチエージェントシステム, 電子商取引システム, Web マイニング, Web 知能化技術の研究に従事。AAAI, 人工知能学会, 電子情報通信学会, 情報処理学会, 日本ソフトウェア科学会各会員。