論 文

Technical Papers

系統的／局所的探索の協調による ファジィ制約充足問題の近似解法

Combining Systematic and Local Search for Approximately Solving Fuzzy Constraint Satisfaction Problems

須藤 康裕
Yasuhiro Sudo
北海道大学大学院工学研究科
Graduate School of Engineering, Hokkaido University
sudoy@main.eng.hokudai.ac.jp

栗原 正仁
Masahito Kurihara
北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University
kurihara@main.ist.hokudai.ac.jp, http://aiwww.main.eng.hokudai.ac.jp

keywords: constraint satisfaction, fuzzy, optimization, iterative improvement, hybrid

Summary

A fuzzy constraint satisfaction problem (FCSP) is an extension of the classical CSP, a powerful tool for modeling various problems based on constraints among variables. Basically, the algorithms for solving CSPs are classified into two categories: the systematic search (complete methods based on search trees) and the local search (approximate methods based on iterative improvement). Both have merits and demerits. Recently, much attention has been paid to hybrid methods for integrating both merits to solve CSPs efficiently, but no such attempt has been made so far for solving FCSPs.

In this paper, we present a hybrid, approximate method for solving FCSPs. The method, called the Spread-Repair-Shrink (SRS) algorithm, combines a systematic search with the Spread-Repair (SR) algorithm, a local search method recently developed by the authors. The SRS algorithm spreads (or expands) and shrinks a set of search trees in order to repair constraints locally until, finally, the satisfaction degree of the worst constraints (which are the roots of the trees) is improved. We empirically show that SRS outperforms the SR algorithm as well as the well-known methods such as Forward Checking and Fuzzy GENET, when the size of the problems is sufficiently large.

1. は じ め に

制約充足問題 (CSP : Constraint Satisfaction Problem) は変数間の制約を全て満たすような変数への値の割当てを解とする組合せ的探索問題である．CSP は NP 完全問題であるため効率的で完全な解法は一般に存在せず，もし，しらみつぶしに解を探索すれば，最悪の場合，計算時間が問題の規模に応じて指数的に増大することは避けられない．しかしながら様々な方法を駆使し，多くの場合に現実的な時間で近似解を得られるよう研究が進められてきた．CSP の解法として大きく 2 つ，部分的な変数割当ての逐次拡張に基づく系統的木探索法と，完全な変数割当ての逐次修復に基づく局所的状態空間探索法とがある．これまで，これらの手法は別個に発展を遂げてきたが，近年両者を効率的に融合したハイブリッド手法が提案され [Jussien 02]，脚光を浴びた (この論文は AAAI-2000 において表彰された論文の改訂版である)．

一方，CSP の単純性は現実問題の定式化を困難にするケースもあることから，必ずしも満たす必要のないソフトな制約を CSP に導入したモデル [Bistarelli 03] についても研究がなされてきている．ファジィ制約充足問題 (ファジィCSP: Fuzzy CSP) もそのようなモデルの一種であり，制約にファジィ関係を導入し，その充足度に曖昧さを持たせることにより，不完全に充足される解を求め，現実の問題解決への有用な情報提供を可能としている [Ruttkay 94, Meseguer 97, Kanada 95]．その解法として，ヒューリスティックを用いた局所的探索法である Spread-Repair アルゴリズムが著者らによって提案され [Sudo 05]，実際に近似解が得られることが種々の問題に対して実証的に示されている．しかしながらその計算効率には改善の余地が残っており，本論文では無駄な探索を取り除くために Spread-Repair アルゴリズムと系統的探索法をハイブリッドに協調させて用いる Spread-Repair-Shrink(SRS) アルゴリズムを提案する．またそ

の評価のためにランダムに生成したファジィCSP の解を求める実験を行い，その近似度および計算時間について他の手法と比較するために，系統的探索法の代表として一般的に高性能であることが知られているフォワードチェック [Haralick 80]，局所的探索法の代表としてニューラルネットを用いた FGENET[Wong 98] および従来の Spread-Repair アルゴリズムを選び，とくに大規模な問題に対する SRS の優位性を示す．

本論文の構成は以下の通りである．2 章では CSP およびファジィCSP について述べる．3 章ではファジィCSP の解法として SRS アルゴリズムを提案する．4 章では評価実験について述べる．5 章では結論を述べる．

2. ファジィ制約充足問題

2·1 制約充足問題

CSP は，変数の集合 $X=\{x_i\}_{i=1}^{n}$ とその値域となる領域の集合 $D=\{D_i\}_{i=1}^{n}$ および制約の集合 $C=\{c_k\}_{k=1}^{r}$ の組として定義される．制約 c_k は X の部分集合 $S_k(S_k \subseteq X)$ 上の関係 R_k であり，S_k に含まれる各変数が同時に取り得る値の組合せを表している．すなわち，$S_k=\{x_{k_1},\cdots,x_{k_w}\}$ ならば $R_k \subseteq D_{k_1}\times\cdots\times D_{k_w}$ である．S_k を R_k の範囲 (scope) という．また，ここで $w=2$ であるとき制約は 2 項制約であるといい，$w=1$ の場合には単項制約，$w\geq 3$ の場合をとくに多項制約という．変数への値の割当てを $v\in D_1\times\cdots\times D_n$ とし，$v[S]$ は割当て v を範囲 S に含まれる変数に制限したもの (射影) とするとき，$v[S_k]\in R_k$ ならば v は制約 c_k を満たし，さもなくば v は制約 c_k に違反する．CSP の解を求めることは，全ての制約を満たす変数への割当てを探し出すことであり，一般に NP 完全問題として知られる．

2·2 ファジィ制約充足問題

ファジィCSP は制約にファジィ関係を導入したモデルであり，最適化問題に位置付けられる．ファジィ制約 c_k はファジィ関係 R_k に対応付けられ，そのメンバーシップ関数 μR_k は

$$\mu R_k : \prod_{x_i\in S_k} D_i \rightarrow [0,1] \qquad (1)$$

の形式で与えられる．すなわち，制約 c_k に対する割当て v のメンバーシップ値 $\mu R_k(v[S_k])$ は，$[0,1]$ の実数値をとり，それが制約の充足度を示す．

制約 c_k と c_l のファジィ論理積 $c_k\wedge c_l$ は，$S_k\cup S_l$ を範囲とするファジィ関係 $R_k\cap R_l$ であり，そのメンバーシップ関数を

$$\mu R_k\cap R_l(v)=min(\mu R_k(v[S_k]),\mu R_l(v[S_l])) \qquad (2)$$

で定義する．同様に全ての制約のファジィ論理積すなわちファジィCSP 全体の充足度を以下のように定義する．

$$\mu\bigcap_{k=1}^{r} R_k(v)=\min_{1\leq k\leq r}(\mu R_k(v[S_k])) \qquad (3)$$

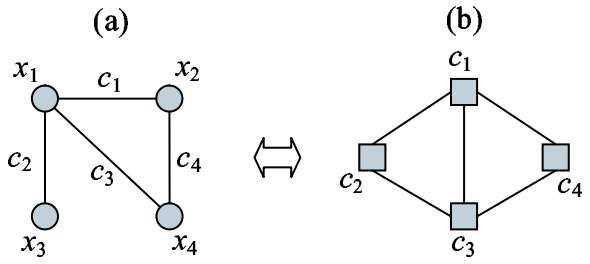

図 1 制約グラフと双対制約グラフ

つまり，全ての制約の中で，最も低い充足度をファジィCSP の充足度 C_{min} とする．

$$C_{min}(v)=\min_{1\leq k\leq r}(\mu R_k(v[S_k])) \qquad (4)$$

ここで $C_{min}(v)>0$ のとき，v をファジィCSP の解という．また，最適解とは最大の充足度を与える解をいう．したがって，ファジィCSP を解くことは，充足度が最低の制約 (c^* とする) の充足度を最大化するような変数への値の割当て v を求める最大化問題となり，以下の式がその目的関数の最大値となる．

$$\max_{v}(\min_{1\leq k\leq r}(\mu R_k(v[S_k]))) \qquad (5)$$

2·3 制約ネットワーク

CSP およびファジィCSP は制約グラフを用いて表現することができる (図 1a)．ノードは変数，辺は制約に対応し，制約 c_k が 2 項制約のとき，その範囲 S_k に属する 2 つのノードを辺で結ぶ．c_k が多項制約のときはその辺を超辺 (端点を 3 個以上持つ辺:hyperedge) とした超グラフ (hypergraph) となる．一方，双対制約グラフ (図 1b) は制約をノードとし，$S_i\cap S_j\neq\phi$ のときにノード c_i，c_j 間に辺をもつグラフであり，辺の端点は必ず 2 個なので超グラフとはならない．

3. ファジィ制約充足問題の解法

通常，CSP およびファジィCSP の変数領域は有限離散集合であり，その要素を列挙することができるので，全ての組合せを探索する木を生成することができる．したがって，しらみつぶしに解を探索することが可能であるが，最悪の場合の計算量は問題のサイズに対し指数的に増大する．CSP の解法は，このような (1) 変数への部分的な割当てを無矛盾性を保ちながら系統的に逐次拡張して完全な割当てを求めようとする系統的探索と，(2) 変数への完全な割当て (すなわち状態) を制約違反が少なくなるように局所的かつグリーディに反復改善することによって状態空間を探索して解を求めようとする局所的探索の 2 つの基本的な分類が挙げられる (表 1)．一般的にはそれぞれ異なる特徴を持ち，これまで別個に研究が進

表 1 2 大探索手法の比較

	系統的探索	局所的探索
基本概念	部分的な割当ての逐次拡張	完全な割当ての局所的修正
探索制御	バックトラック法による深さ優先探索を基本とし，フォワードチェックやアーク整合などの領域フィルタリングや経路整合などの制約推論による枝刈りで効率化をはかる	ヒューリスティックな評価関数の値を反復的に改善するために，グリーディに 1 つの変数の値を修正する．局所最適解からの脱出方法は別途設計する．
完全性	完全性あり (解が存在すれば必ず発見する)	通常，完全性をもたない*1．制約違反をできるだけ少なくした近似解を得ることができる．その過程で正しい解が見つかることも多い．
探索効率	大規模で複雑な問題では解を得るまでに長時間を要することが多いとされている	大規模で複雑な問題でも (解が得られるときには) 短時間で済む例が数多く報告されている

められてきたが，近年は，両者を効率的に統合した新しい手法が提案され注目を浴びた [Jussien 02].

ファジィCSP に関する研究では，これまで分枝限定法 [Meseguer 97] やニューラルネット [Wong 98, Adorf 90] など様々な方法で解が得られることが示されている．Spread-Repair アルゴリズム [Sudo 05] は連続領域を含むファジィCSP を解くために提案された手法であるが，離散領域のファジィCSP の近似解も求めることができる．このアルゴリズムは局所的探索法の一種であるが，本章ではその一部に系統的探索手法を導入し，効率を高めた新しいハイブリッド手法を提案する．

3·1 関 連 研 究

§1 ファジィCSP の系統的探索法

ファジィCSP においてはファジィ論理積が最小値を意味することから，系統的探索におけるバックトラックを効果的に用いることができ，同時に緩和問題から得られる上界 (下界) を利用した分枝限定法を適用することができる．文献 [Meseguer 97] では，一般的に効率の良いことが知られるフォワードチェック [Haralick 80] と呼ばれる手法をファジィCSP に適用できることを示し，評価実験を行っている．ただし CSP の性質として一般的にはどんなに効率的に枝刈りを行ったとしても，最悪の場合，問題の規模に応じて計算量の指数的増加は避けられないが，厳密解が必要な場合には分枝限定法は有効な手法であり，比較的規模の小さい問題においては実用的である．

§2 ファジィCSP の局所的探索法

系統的探索に対し，大規模な問題に対する局所的探索法の平均的な効率の良さが知られている [Adorf 90]．Fuzzy GENET(FGENET)[Wong 98] はニューラルネットモデルの一種であり，2 項制約を持つファジィCSP の解を得るための手法である．クラスタは変数に相当し，変数領域の要素と同じ数のニューロンを持つ．制約が存在するクラスタ間のニューロン同士をすべて接続し，そこに充足度に応じた重みを設け，入力に応じてニューロンの値を変更する．その操作は非同期に行われ，局所最適において重みを変化することにより局所解からの脱出を行う．非常に単純なアルゴリズムであるが，文献 [Bowen 96] で示された手法と同等の結果が得られている．

Chemical Casting Model(CCM)[Kanada 95] は化学反応における自己組織系をモデルにした局所的探索法の一種であり，局所的な情報のみによって確率的に動作するアルゴリズムである．並列分散処理が可能であること，プログラムが単純小規模で済むなどの利点がある反面，全解を得るのが困難であることや，収束にかかる時間が大きいといった欠点もある．

§3 ハイブリッドな探索

従来の CSP に関するハイブリッド手法の研究において，文献 [Jussien 02] では系統的探索と局所的探索の融合には以下の 3 つの考え方があるとしている．

- 局所的探索を系統的探索の前または後に行う
- 系統的探索における枝 (部分的な割当て) または葉 (完全な割当て) において局所的探索を行う
- 全体を通して局所的探索を行い，探索空間における近傍から 1 つの次状態を選択する際または探索空間の枝刈りに系統的探索を用いる

いずれにせよ局所的探索法の持つ大規模な問題への有効性と，系統的探索法の持つ完全性というそれぞれの長所をどのように生かすかが重要になる．近年ではスケジューリング等のオペレーションズ・リサーチに関する研究でもハイブリッド手法に関する研究 [Lim 04] がある．ファジィCSP に関するハイブリッド手法の研究では，遺伝的アルゴリズム (GA) と系統的探索における領域フィルタリング (アーク整合) の併用 [Bowen 96] があるが，[Wong 98] のような典型的な局所的探索の効率を超えるものではない．また，GA を使うので局所的探索とは言えず，ここで論ずるハイブリッド手法とは少々離れることから，本論文では比較の対象とはしないものとする．

*1 no-good データベースやタブーリストを学習させて不適切な一部の状態への訪問を回避することによって完全性をもたせることもできる

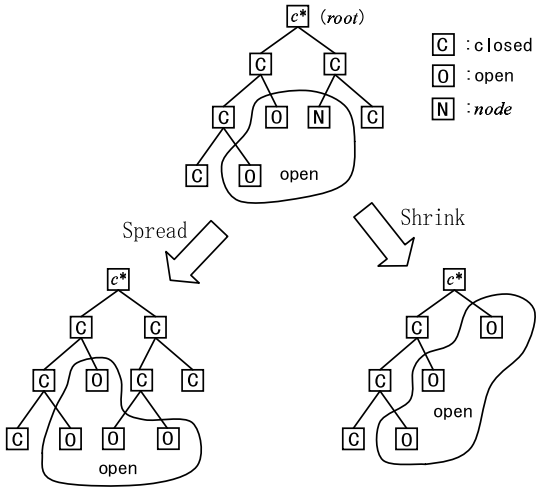

図 2 SRS における探索木 (open/closed ノード)

3·2 SRS アルゴリズム

本節ではファジィCSP を効率的に解くためのハイブリッドな手法，Spread-Repair-Shrink(SRS) アルゴリズムを提案する．基本的な構造は前節の 3 つ目の考え方に似ているが，そのどれにも属さず，

- 全体を通して局所的探索を行い，一連の逐次改善を局所的に制御するために系統的探索を用いる

という考え方が基底となっている．

§1 基本的な動作

(5) 式の定義より，ファジィCSP は最悪の充足度の最大化を目的関数とする．Spread-Repair(SR) アルゴリズムは最悪の充足度をもつ制約 c^* の改善を繰り返し，局所最適状態において改善の対象を近傍の制約に拡張する (*Spread* と呼ぶ) ことにより局所解を抜け出す局所的探索アルゴリズムである．この手法の特徴は，c^* を改善していく過程において，全ての制約充足度の平均

$$C_{ave} = \frac{1}{r}\sum_{k=1}^{r} \mu R_k(v[S_k]) \qquad (6)$$

も同時に増加させることができる点である．現実問題として，平均的な充足度を軽視することは応用面で不利益を生み，これまでの多くの研究で評価の対象となっている [Ruttkay 94, Meseguer 97, Kanada 95].

SRS アルゴリズムは，SR アルゴリズムにおける改善対象の拡張の際に生じる無駄な判定を，一種の探索木を用いた効率的なバックトラック (SRS においては *Shrink* と呼ぶ) を行うことにより抑制し，計算時間の短縮を実現するアルゴリズムである．SR では c^* を単体として扱っていたが，一般に c^* は複数存在し，SRS はそれらを根とする複数の探索木 (森) を双対制約グラフの上で成長あるいは縮小させながら近傍の制約を改善しつつ，最終的にすべての c^* を改善しようとする．基本構造はグラフ探索でよく知られる open/closed リストを用いたアルゴリズムに基づいており，以下で示す基本的な 3 つのモードを切り替えながら探索を行う．

§2 Repair

探索木のノードは制約を表しており，その状態は，open か closed かのいずれかで，それぞれ open リストと closed リストに格納されている．open ノードは局所的な改善を試みようとしているノードであり，木の終端ノードの集合の部分集合になっている．その他の終端ノード及び木の非終端ノードの全ては closed ノードであり，局所的に改善不可能であることが既知であるノードである (図 2)．Repair モードは open ノードのうちの 1 つ (以下 *node* と記し，対応する制約を一般性を失うことなく c_0 とする) に着目し，制約 c_0 を改善可能かどうかチェックし，可能であれば実際に改善を行う．ここで制約 c_0 の改善 (Repair) を以下のように定義する．

【定義 1】(Repair) 範囲 S_0 の要素である 1 つの変数への割当てを変更し，c_0 の充足度を高める．

$$\textbf{select } v'$$

$$\textbf{subject to } \mu R_0(v'[S_0]) > \mu R_0(v[S_0])$$

v は現在の割当て，v' は変更後の割当てである．ただし，そのような割当て v' を 1 つ決定する (状態空間における近傍を決定する) 場合，複数の候補から何らかの意味で局所的に最適なものを選択することが望ましいが，アルゴリズムの性質を限定しないためにここでは抽象的な手続きとして Repair を定義した．以下にいくつか実装例を示す.

改善したい制約を c_0，その近傍 (それに隣接した制約) を $c_1, c_2, \cdots, c_m$ とする．最も貪欲な考えは，近傍の充足度を下げることなく c_0 の充足度を最大化することである．このアルゴリズムを SRS1 と呼ぶ.

$$\textbf{maxmize } \mu R_0(v'[S_0])$$

$$\textbf{subject to } \mu R_0(v'[S_0]) > \mu R_0(v[S_0])$$

$$\mu R_i(v'[S_i]) \geq \mu R_i(v[S_i]), 1 \leq i \leq m$$

ただしこのような候補が見つかることは比較的少なく，たちまち局所最適に陥る場合がほとんどである．SRS1 は非常に短時間でアルゴリズムが停止しその保証も得られる[*2]が，解の質は比較的劣悪である．

次に，SR で用いている考え方を示す．これはファジィ論理積が最小を意味することから，局所的な改善においても最小の充足度を最大化するものであり，SRS2 と呼ぶ.

$$\textbf{maxmize } \min_{0 \leq i \leq m} (\mu R_i(v'[S_i]))$$

$$\textbf{subject to } \mu R_0(v'[S_0]) > \mu R_0(v[S_0])$$

SRS2 では局所的な鞍点を求めることと同様であり，その結果 c_0 の改善が控えめな傾向となる．しかし予備実験

*2 アルゴリズムの停止性に関して，SRS1 は貪欲な戦略であるため必ず停止する．SRS2，SRS3 はいわゆる”高原”において巡廻に陥る可能性がある

では SRS2 は SRS1 よりも解の質が大幅に向上することが確認できている．

最後に c_i の充足度が $C_{min}(v)$ より小さくならない範囲で最も c_0 の充足度が増加するような v' を選択する SRS3 を示す．

$$\textbf{maxmize}\ \ \mu R_0(v'[S_0])$$

$$\textbf{subject to}\ \ \mu R_0(v'[S_0]) > \mu R_0(v[S_0])$$

$$\mu R_i(v'[S_i]) \geq C_{min}(v), 1 \leq i \leq m$$

予備実験では SRS3 では SRS2 と同等以上の解の質が得られることが確認できている．

このようにさまざまな考え方を適用することができるが，これらは問題の性質や状況等によって使い分けることが好ましい．ただし，4 章の評価実験のテスト問題においては，解の質と計算時間，実装のコスト等から総合的に優れていると判断した SRS3 を用いている．

§ 3 Spread

SRS において，通常のグラフ探索アルゴリズムにおける目標ノードに相当するのが，1 変数の割当て変更によって局所的に改善可能である制約を表すノードである．SRS は c^* を改善できないときに，その近傍の制約の改善を繰り返すことによって，その影響を c^* に伝播させて c^* を改善できる状態に移行させようと試みる．その周辺の探索過程において探索木を成長 (Spread) させるのが Spread モードであり，open ノード *node* を改善できないときはそれを closed ノードとし，それに隣接する open でも closed でもないノードを新たに open ノードとし，これらを *node* の子ノードとして探索木に付加する．すなわち，これはグラフ探索アルゴリズムで良く知られる *node* の展開操作[*3]である (図 2)．

§ 4 Shrink

ある制約 (ノード) が改善された場合，アルゴリズムは Shrink モードに切り替わり，改善すべき位置を木の根である c^* に近づけるために木を縮小 (Shrink) させる．*node* をその親 (parent) ノードに置き換え，新たな *node* の子ノードはすべて open/closed リストから削除し，再帰的に Shrink を行う．SR アルゴリズムでは Spread した後，再びしらみつぶしに *root*(すなわち c^*) から改善を試みるが，SRS では *node* の改善の影響を効率良く伝播させるために Spread してきた経路を木構造として動的に管理している (図 2)．

§ 5 SRS アルゴリズム

図 3 に SRS アルゴリズムを示す．*open*，*closed* は順序集合 (リスト) とし，要素の追加および削除においては後に述べるヒューリスティックを用いてその順序を決定する．parent，children，descendants は探索木における構造に対応する親，子，子孫を表す．以下に手続きの概要を示す．

*3 ただし英単語としては expand ではなく spread とする

```
Function SRS(constraints C)
    {input C: a set of the worst constraints}
    closed ← φ.
    open ← C.
    While C ≠ φ and open ≠ φ do
        node ← the first element of open.
        open ← open \ node.
        If not Repair(node) then
            Spread()
        Else if node ∈ C then
            C ← C \ {node}.
        Else if Repair(the parent of node) then
            Shrink( )
        Else
            Spread( )
        End if
        Sort open by heuristic h( ).
    End while
    If C = φ then return TRUE
    Else return FALSE
End function

Procedure Spread( )
    closed ← closed ∪ node.
    open ← open ∪ neighbors(node) \ closed.
End procedure

Procedure Shrink( )
    node ← parent of node.
    open ← open \ the descendants of node.
    closed ← closed \ descendants of node.
    If node ∈ C then
        C ← C \ {node}.
    Else
        open ← open ∪ node.
        If Repair(the parent of node) then Shrink( )
    End if
End procedure

Function Repair(constraint c)
    Try to repair c.
    If repaired then
        return TRUE
    Else
        return FALSE
    End if
End function

Begin program
    Do while SRS(worst constraints).
End program
```

図 3 SRS アルゴリズム

- メインプログラムは，c^* の集合 C を求めては関数 SRS(C) を呼び出すという処理を SRS が `FALSE` を返すまで繰り返す．
- 関数 SRS(C) は，c^* の集合 C を受け取り，すでに述べた考え方に基づいて Spread()，Repair(*node*)，Shrink() の各手続きを適切に呼び出して制御し，C のすべてが改善できたら `TRUE` を返し，失敗したら `FALSE` を返す．
- 手続き Spread() は Spread モードに入り，探索木の拡張を行う．
- 手続き Shrink() は Shrink モードに入り，探索木の縮小を行う．再帰的に *node* の親ノードである制約の改善を試みる．
- 手続き Repair(c) は，制約 c の Repair を試みる．成功すれば `TRUE`，さもなければ `FALSE` を返す．

表 2 結果 (1): 解の質に関する実験

Algorithms	d	$t=0.2$		$t=0.4$		$t=0.6$		$t=0.8$		ratio
		C_{min}	C_{ave}	C_{min}	C_{ave}	C_{min}	C_{ave}	C_{min}	C_{ave}	
SRS3	0.2	0.865	0.942	0.704	0.837	0.485	0.618	0.316	0.489	1(3)
FC		0.943	0.969	0.808	0.879	0.592	0.669	0.453	0.550	19000
SRS3	0.4	0.781	0.895	0.591	0.733	0.404	0.546	0.213	0.380	1(6)
FC		0.850	0.911	0.670	0.760	0.468	0.563	0.278	0.408	13000
SRS3	0.6	0.740	0.865	0.551	0.703	0.352	0.520	0.168	0.336	1(14)
FC		0.800	0.889	0.616	0.728	0.717	0.537	0.236	0.364	8200
SRS3	0.8	0.708	0.853	0.525	0.683	0.330	0.482	0.156	0.326	1(26)
FC		0.767	0.866	0.579	0.714	0.386	0.522	0.206	0.338	7100

§6 ヒューリスティック等

open，*closed* 内の順序に関して，さまざまなヒューリスティックが考えられる．単純な影響度を予測すれば，(1)c^* までの距離が小さい，すなわち探索木の階層が浅い制約の優先順位を高くする方法が考えられるが，(2) 制約充足度の昇順にしたり，(3) 変数割当ての変化量で降順にするなどさまざまなアイデアがあり，問題の性質等によって経験的に決定すべきである．ただし 4 章の実験では (1) を用いることとする．

4. 実　　験

SRS アルゴリズムの性能を評価するため，いくつかの実験を行う．大別して，得られる解の質と，計算時間の 2 点について，系統的探索法と局所的探索法との比較を行う．前者の代表として，一般的に低コストで効率の良さが知られているフォワードチェック [Haralick 80] を用いる．文献 [Meseguer 97] では *LB2* として評価実験が行われており，制約伝播法 (constraint propagation) を用いた分枝限定法の一種といえる．後者の代表としては SR アルゴリズムおよびニューラルネットを用いた手法の一種である Fuzzy GENET[Wong 98] を用いる．解の質の評価には，C_{min} および C_{ave} の 2 つの値を用い，計算時間では実装による実時間 (ms) を用いる．

ここでは SRS3 を選択し，さらに SRS の設定でヒューリスティックとした *open* 内の順序を前節の (1)，すなわち *root* からの距離 (探索木の階層) で昇順とする．また，同レベルの要素は非決定的に選択するものとする．この設定においては SRS の停止性が保証されないが，以後の実験では全ての問題例で計算の停止が確認できたので，SRS における全ての結果は計算を途中で打ち切ったものではなく，アルゴリズム自体の停止条件によって計算が終了したときの充足度および時間である．

4·1 系統的探索 FC との比較

ファジィCSP は最大化問題の一種であるため，得られる解には充足度が付随する．一般的に最適化問題における最適解を得るのは困難であるが，現実世界では厳密な最適解が必ずしも必要とされない．近似解とは一般に最適解とは限らない解のことであり，近似度とは最適解に対する近似解の目的関数値の比である．ここでは解の質 (近似度) を比較するための実験を行う．

評価に用いる問題はランダムに生成した 2 項制約 ($w=2$) のファジィCSP であり，パラメータ $\{n,d,t\}$ で与えられ，それぞれ変数の数，制約ネットワークの密度 (density)，制約の平均緊密度 (tightness) を表す [Culberson 01]．ここでは，d および t を以下のように定義する．

$$d=\frac{r}{(n^2-n)/2} \tag{7}$$

$$t=\frac{1}{r}\sum_{k=1}^{r}\left[\frac{1}{\prod_{i=1}^{w}|D_i|}\cdot\sum_{v\in\prod_{i=1}^{w}D_{k_i}}\mu R_k^{-1}(v)\right] \tag{8}$$

ただし $S_k=\{x_{k_1},\cdots,x_{k_w}\}$．すなわち，$d$ は完全グラフとの辺の数の比，t はあらゆる変数割当ての組合わせにおける各制約違反度の平均とする．実験では $n=10$，変数領域のサイズ $\delta=|D_1|=\cdots=|D_n|=10$，$d$，$t$ をそれぞれ {0.2, 0.4, 0.6, 0.8} とし，4 × 4=16 クラスの問題をそれぞれ 100 個生成し，C_{min} および C_{ave} における評価を平均する．

表 2 の FC はフォワードチェックを示す．また ratio は SRS が要した計算時間との比を示し，括弧中は実測値 (ms) の平均である．FC で得られた解は最適解であり，その結果と比較すると C_{min} に関して $t=0.2$ においては最適解に対する近似度が 90%を超える結果が得られており，平均でも 80%以上の近似度が得られている．C_{ave} に関しても，FC で得られた結果に対して C_{min} とほぼ同等の近似度が得られている．

4·2 局所的探索 SR および FGENET との比較

最適化問題の近似解法の性能評価においては，解の質と共に (局所) 最適解への収束の速度が問題となる．図 4 は時間経過に対する充足度の収束を示すグラフである．CSP においては一般に，制約密度などのパラメータの変化によって問題の性質が大きく変わり [Hogg 96]，計算量に大きな差が生じる場合がある．とくに問題の種類によっては一部のパラメータで非常に難しい問題例が現れ

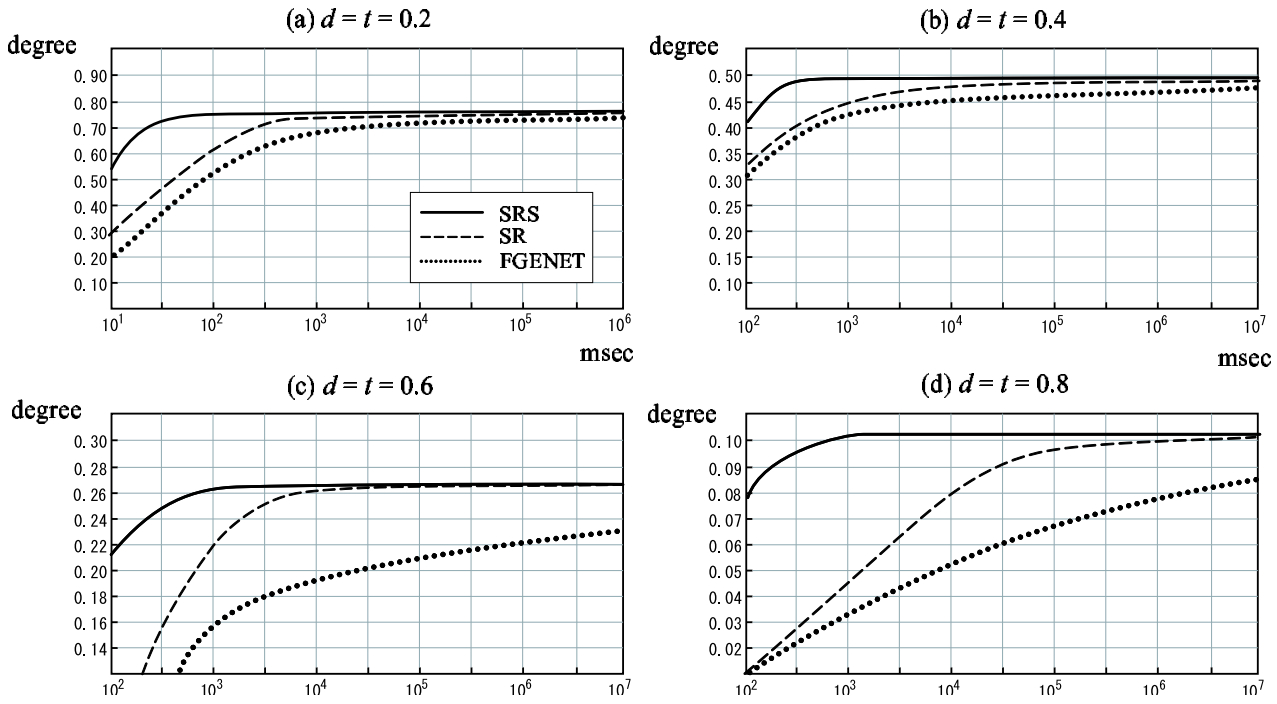

図 4 時間経過に対する充足度の収束

ることが知られている．ファジィCSP に関しては実験 (1) の結果および文献 [Wong 98] の結果より，とくにそのような現象は確認できない．したがって図 4 ではパラメータ d および t を 0.2，0.4，0.6，0.8 に限定して設定した結果のみを示している．問題のサイズは $n = 20$，$\delta = 10$ である．

全ての結果において，とくに初期状態から直後の収束に非常に大きな差がみられる．また，問題のパラメータに関する収束速度の差はそれほど大きくはないが，問題が難しく (最適解の充足度が低く) なるにしたがって FGENET の収束が遅くなる傾向が見られる．その理由としては，制約が密になり局所解が多数存在する状況においては，FGENET の重み計算が頻発し，計算量が増大するためと考えられる．

これらの収束速度の差は，実装の詳細による影響を遥かに上回ると考えられ，以上の結果より，とくに現実時間で近似解を求めたい局面において，SRS は FC および FGENET と比較して良く性能を発揮するといえる．

4·3 各アルゴリズムの漸近特性

それぞれのアルゴリズムにおいて，問題のサイズに対する計算時間の増加傾向を図 5 に示す．(a) は横軸に変数のサイズ n，(b) は変数領域のサイズ δ，縦軸はそれぞれ計算時間 (ms) を表し，片対数グラフになっている．FGENET の計算時間は，SRS と同等の充足度をもつ解が得られるまで計算した場合の時間 (ms) である．

(a)，(b) 共に直線の傾きで表される指数関数の基数部分が異なっており，他の 3 つのアルゴリズムに対する SRS の問題規模に対する優位性があるといえる．

5. ま と め

本論文ではファジィCSP の近似解を効率的に得るためのアルゴリズムとして系統的／局所的探索をハイブリッドに協調して用いた Spread-Repair-Shrink を提案し，ランダムに生成した問題例を用いて系統的探索法の 1 つとして FC，および局所探索法の代表として SR および FGENET との比較を行った．その結果，とくに問題の規模に対する計算時間の増加傾向に関して相対的に効率が良いことが示された．おおまかに言って，FC を用いれば厳密解を発見することができ，FGENET を十分長時間走らせればSRS より質の良い解が得られるという特徴*4に対して，SRS では非常に短時間で実用的な近似度の解を得ることができるという特徴を持つ．

SRS は他のアルゴリズムの前処理のためのフィルタリングアルゴリズムとしても活用できる．たとえば，SRS の近似解の充足度を FC の枝刈りのための初期値として用いた実験では FC の計算速度は数百から数千倍高速になり，インスタンス間における計算時間のバラつきが非常に小さくなった．

近年ファジィCSP を包含するソフト CSP と呼ばれる抽象的なクラスの問題に対し，半環に基づいた定式化と制約伝播を用いた解法の研究が進んでいる．今後の課題として，そのような問題に対しても提案手法を用いて近

*4 FGENET は基本的に停止しないアルゴリズムであるため，最終的には最適解に近付く可能性がある

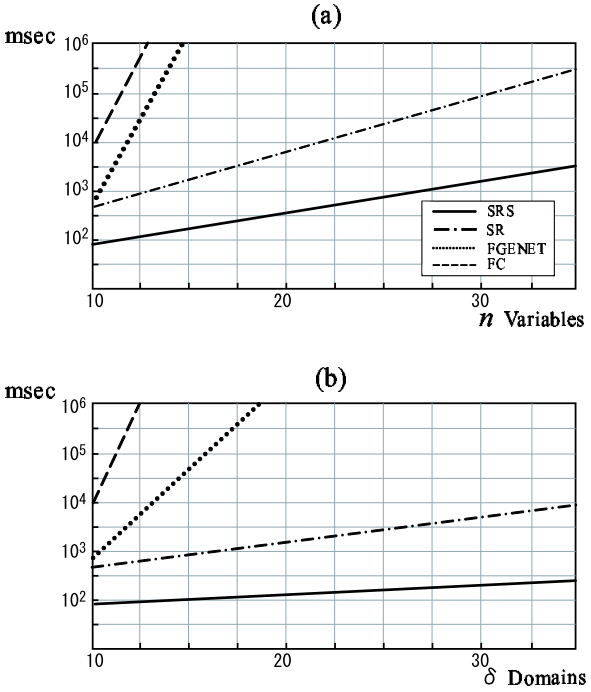

図 5 問題のサイズに対する計算時間の変化

似解を求めることができるよう，アルゴリズムを一般化していくことが挙げられる．

◇ 参 考 文 献 ◇

[Adorf 90] Adorf, H.M., Johnston, M.D.: A discrete stochastic neural networks algorithm for constraint satisfaction problems, *Proceedings International Joint Conference on Newral Networks*, CA(1990)

[Bistarelli 03] Bistarelli, S.: Semirings for Soft Constraint Solving and Programming (Lecture Notes in Computer Science 2962), SpringerVerlag(2004)

[Bowen 96] Bowen, J., Dozier, G.: Solving Randomly Generated Fuzzy Constraint Networks Using Evolutionary/Systematic Hill-Climbing, *Proceedings of 5th IEEE Intern. Conf. on Fuzzy Systems*, vol.1, pp.226-231(1996)

[Culberson 01] Culberson, J., Walsh, T.: Tightness of Constraint Satisfaction Problems, APES-29(2001)

[Dechter 03] Dechter, R.: Constraint Processing, Morgan Kaufmann(2003)

[Fox 87] Fox, M.S.: Constraint-Directed Search: A Case Study of Job-Shop Scheduling, Morgan Kaufmann, San Mateo, CA(1987)

[Haralick 80] Haralic, R.M., Elliot, G.L.: Increasing Tree Search Efficiency for Constraint Satisfaction Problems, *Artificial Intelligence*, Vol.14, pp.263-313(1980)

[Hogg 96] Hogg, E., Bernardo, A., Huberman, C., Williams.: Phase transitions and the search problem, *Artificial Intelligence*, Vol.81, Issues1-2, pp.1-15(1996)

[Jussien 02] Jussien, N., Lhomme, O.: Local search with constraint propagation and conflict-based heuristics, *Artificial Intelligence*, Vol.139, pp.21-45(2002)

[Kanada 95] Kanada, Y.: Fuzzy Constraint Satisfaction Using CCM - A Local Information Based Computation Model, *Proceedings of 4th IEEE Intern. Conf. on Fuzzy Systems*, Vol4, pp.2319-2326(1995)

[Lim 04] Lim, A., Rodrigues, B., Thangarajoo, R., Xiao, F.: A Hybrid Framework for Over-Constrained Generalized Resource-Constrained Project Scheduling Problems, *Artificial Intelligence Review*, 22, pp.211-243(2004)

[Meseguer 97] Meseguer, P. and Larrosa, J.: Solving fuzzy constraint satisfaction problems, *Proceedings of 6th IEEE Intern. Conf. on Fuzzy Systems*, Vol3, pp.1233-1238(1997)

[Minton 90] Minton, S., Johnston, M.D., Philips, A.B., Laird, P.: Solving Large-Scale Constraint Satisfaction and Scheduling Problems Using a Heuristic Repair Method, *Proceedings of AAAI-90*, Vol.1, pp.17-24(1990)

[Minton 92] Minton, S., Johnston, M.D., Philips, A.B., Laird, P.: Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems, *Artificial Intelligence* 58, pp.161-205(1992)

[西田 99] 西田豊明：人工知能の基礎, 丸善株式会社 (1999)

[Ruttkay 94] Ruttkay, Zs.: Fuzzy constraint satisfaction, *Proceedngs of 3rd IEEE Intern. Conf. on Fuzzy Systems*, Vol3, pp.1263-1268(1994)

[Sudo 05] Sudo, Y., Kurihara, M., Mitamura, T.: Spread-Repair Algorithm for Solving Extended Fuzzy Constraint Statisfaction Problems, *Proceedings of the 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology*, pp.891-901(2005)

[William 88] William, H.P., Brain, P.F., Saul, A.T., William, T.V.: NUMERICAL RECIPES in C, Cambridge University Press(1988)

[Wong 98] Wong, J.H.Y. and Leung, H.: Extending GENET to Solve Fuzzy Constraint Satisfaction Problems, *Proceedngs of 15th National Conf. on Artifitial Intelligence*(*AAAI*-98), pp.380-385(1998)

〔担当委員：櫻井 彰人〕

2005 年 7 月 7 日 受理

著 者 紹 介

須藤 康裕（学生会員）

2003 年北海道工業大学大学院工学研究科 電気工学専攻修士課程修了．2003 年北海道大学大学院工学研究科 システム情報工学専攻博士後期課程に入学，現在に至る．人工知能に関する研究に従事．日本知能情報ファジィ学会，情報処理学会，電子情報通信学会各会員．

栗原 正仁（正会員）

1980 年北海道大学大学院工学研究科情報工学専攻修士課程修了．同年北海道大学工学部助手．以後，講師，助教授を経て，1996 年北海道工業大学教授．2002 年北海道大学教授，現在に至る．工学博士．人工知能およびソフトウェア科学の研究に従事．情報処理学会，電子情報通信学会，米国人工知能学会各会員．