2F3-06

ラグランジュの方法を用いたCSPの解法

Solving CSP by Lagrangian Method

中野隆宏　　永松 正博
Takahiro Nakano　Masahiro Nagamatu

九州工業大学大学院 生命体工学研究科
Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology

The constraint satisfaction problem(CSP) is a combinatorial problem to find a solution which satisfies all given constraints. We proposed a neural network called LPPH to solve the satisfiability problem(SAT). Each equilibrium point of the LPPH is a solution of the SAT, and LPPH is not trapped by local minima. In this paper,we extend the LPPH for solving the CSP. The CSP can represent problems more simply compared with the SAT. Experiment results show our proposal method is effective.

1. はじめに

与えられた制約を全て満足するような変数の値の割り当てを探索する問題を制約充足問題 (CSP;Constraint Satisfaction Problem) という．CSP は，問題の抽象性が高く，様々な問題を表現することができる．また，CSP に解が存在するかどうかという判定問題は NP 完全問題であることが知られている．我々は充足可能性問題 (SAT;SATisfiability problem) に対し，LPPH と呼ばれる手法を提案している [Nagamatu 1995]．本論文では CSP を解くため，LPPH を拡張する方法を提案する．スケジューリング問題などの現実の組み合わせ問題を SAT に変換する場合，SAT では制約を和積標準形で表現するため，問題の規模が大きくなるにしたがって節の個数が膨大なものにってしまう．CSP では制約をより一般的な形で扱えるため，SAT に比べてよりコンパクトに問題を表現することができる．また，実験により提案手法の有効性の検証を行う．

2. CSP

CSP とは，与えられた制約を全て満足するように変数に値を割り当てる問題である．CSP は次のような (X, D, C) で定義される．

- $X = \{X_1, X_2, X_3, \cdots, X_n\}$ は変数の有限集合．
- $D = \{D_1, D_2, D_3, \cdots, D_n\}$ は離散値を要素とする変域の有限集合であり，変数 x_i は変域 D_i の要素のひとつを値としてとる．
- $C = \{C_1, C_2, \cdots, C_m\}$ は CSP の制約の集合．

C を全て満たすような X への値の割り当てが CSP の解となる．

3. CSP の制約

より効率的に CSP を表現するために，一般的な制約を考える．X_i $(i = 1, 2, \cdots, n)$ と変域 D_i $(i = 1, 2, \cdots, n)$ における値 d_j $(j = 1, 2, \cdots, l_j)$ のペアを VVP(Variable-Value Pair) と呼ぶ．また，VVP の要素 (X_i, d_j) を x_{ij} と書く．要素 X_i が値 d_j をとることを x_{ij} が真となるということにする．制約 C_r は VVP の集合であり，含まれる x_{ij} の内，少なくとも 1 つが真でなければならないというという制約を ALOT(At Least One True) 制約と呼ぶ．同様にして，ALOF(At Least One False) 制約，ATMOST(At Most One True) 制約，AMNT(At Most N True) 制約などが考えられる．

瞳: 永松 正博，九州工業大学大学院 生命体工学研究科，〒 808-0135 北九州市若松区ひびきの 2-4，tel/093-695-6088，fax/093-695-6088，nagamatu@brain.kyutech.ac.jp

4. ラグランジュの方法による解法

VVP の集合を $\boldsymbol{x} = \{x_{11}, x_{12}, \cdots, x_{nl_n}\}$ とする．以下 x_{ij} は変数 i が値 j をとる確かさを表す変数と考え，0 から 1 の実数値をとるとする．すなわち，$x_{ij} = 1$ ならば，変数 i に値として j が割り当てられ，$x_{ij} = 0$ ならば，変数 i に値として j が割り当てられないことを意味する．

本論文では，CSP の解法として，次のような力学系をもつ手法 LPPH-CSP を提案する．

$$\frac{dx_{ij}}{dt} = x_{ij}(1 - x_{ij}) \sum_{r=1}^{m} w_r s_{rij}(\boldsymbol{x})$$

$$\frac{dw_r}{dt} = -\alpha w_r + h_r(\boldsymbol{x})$$

上式において，関数 $s_{rij}(\boldsymbol{x})$ は制約 r を充足させるため x_{ij} が変化させようとする力を表す．$h_r(\boldsymbol{x})$ は制約 r が充足していない度合いを表す関数であり，制約 r が充足したなら 0 を返し，制約 r が充足していないなら正の値を返す関数である．α は減衰係数とよばれ，解の探索の効率に大きな影響を与える．LPPH-CSP において各変数は制約を充足させるように変化し，制約の重み w_r は制約 C_r が充足していないならば大きくなる．LPPH-CSP は上記の連立微分方程式を解くことにより，CSP の解を探索する．

3 節で示した制約の $h_r(\boldsymbol{x}), s_{rij}(\boldsymbol{x})$ として 2 つの手法を提案する．1 つは，充足していない度合いに多重 1 次関数を使用する LPPH-CSP1 であり，もう 1 つは充足していない度合いに Min/Max 関数を使用する LPPH-CSP2 である．各手法における ALOT 制約の $h_r(\boldsymbol{x}), s_{rij}(\boldsymbol{x})$ を次に示す．

< LPPH-CSP1 (ALOT 制約) >

$$h_r(\boldsymbol{x}) = \prod_{\substack{0 \le i \le n \\ 1 \le j \le l_n}} (1 - f_{rij}(\boldsymbol{x}))$$

$$s_{rij} = -\frac{\partial h_r(\boldsymbol{x})}{\partial x_{ij}}$$

$$g_{rij}(\boldsymbol{x}) = \begin{cases} x_{ij} & \text{if } x_{ij} \text{ is in } C_r \\ 0 & \text{otherwise} \end{cases}$$

< LPPH-CSP2（ALOT 制約）>

$$
\begin{aligned}
h_r(\boldsymbol{x}) &= 1 - \mathrm{Max}\{g_{rij}(\boldsymbol{x}) | 0 \le i \le n,\ 1 \le j \le l_n\} \\
s_{rij}(\boldsymbol{x}) &= 1 - \mathrm{Max}\{g_{rst}(\boldsymbol{x}) | 0 \le s \le n,\ 1 \le t \le l_n,\ st \ne ij\}
\end{aligned}
$$

各手法の ALOF 制約の $h_r(\boldsymbol{x}), s_{rij}(\boldsymbol{x})$ は次のようになる．
< LPPH-CSP1（ALOF 制約）>

$$
\begin{aligned}
h_r(\boldsymbol{x}) &= \prod_{\substack{0 \le i \le n \\ 1 \le j \le l_n}} f_{rij}(\boldsymbol{x}) \\
s_{rij} &= -\frac{\partial h_r(\boldsymbol{x})}{\partial x_{ij}} \\
f_{rij}(\boldsymbol{x}) &= \begin{cases} x_{ij} & \text{if } x_{ij} \text{ is in } C_r \\ 1 & \text{otherwise} \end{cases}
\end{aligned}
$$

< LPPH-CSP2（ALOF 制約）>

$$
\begin{aligned}
h_r(\boldsymbol{x}) &= \mathrm{Min}\{f_{rij}(\boldsymbol{x}) | 0 \le i \le n,\ 1 \le j \le l_n\} \\
s_{rij}(\boldsymbol{x}) &= -\mathrm{Min}\{f_{rst}(\boldsymbol{x}) | 0 \le s \le n,\ 1 \le t \le l_n,\ st \ne ij\}
\end{aligned}
$$

ここでは ALOT 制約，ALOF 制約の $h_r(\boldsymbol{x}), s_{rij}(\boldsymbol{x})$ を述べたが，3 節に示した他の制約も同様も同様に表現することができる．

5. 実験

5.1 提案手法の比較実験

4 節で示した LPPH-CSP1,LPPH-CSP2，CSP を SAT に変換し従来の LPPH を用いる手法について比較実験した．各手法について初期値を 30 回変え，N-Queen 問題の解を探索するのに要した CPU 時間の平均を表 1 に示す．

表 1: CPU 時間の比較

問題	LPPH	LPPH-CSP1	LPPH-CSP2
10-Queen	0.160 s	0.044 s	0.010 s
20-Queen	3.537 s	2.206 s	0.030 s
30-Queen	21.484 s	9.903 s	0.150 s
40-Queen	66.512 s	26.803 s	0.360 s
50-Queen	120.657 s	78.070 s	0.542 s

実験結果より LPPH-CSP2 が効率よく解を探索できることがわかる．

5.2 減衰項に関する実験

LPPH-CSP2 における減衰係数 α に関して実験を行った．減衰係数のパラメータによって解の探索効率がどう変化するか調べるため，減衰係数の値を変化させ，それぞれの減衰係数の値での解の探索に要した平均 CPU 時間を求めた．Car Sequencing，Random CSP を解いた結果を図 1，2 に示す．Car Sequencing 問題は，CSPLib(www.csplib.org) のベンチマーク問題 (変数の個数 200) を用いた．Random CSP は変数の個数を 20，変域の値の個数を 10 とし，任意に変数間の 2 項制約を作成することにより生成した．図 1，2 において縦軸は平均 CPU 時間，横軸は横軸は減衰係数 α，系列は用いた問題を表している．図 1，2 の実験結果より，Car Sequencing 問題および Random CSP においては $\alpha = 0.1$ 付近が最も効率的に解を探索していることが確認できる．

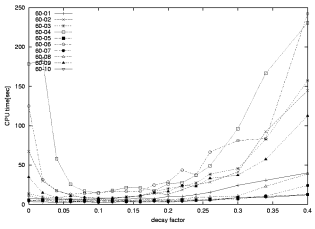

図 1: Car Sequencing 問題

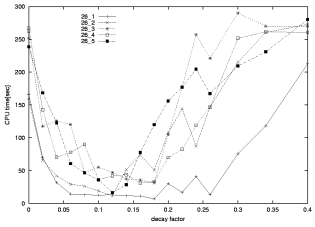

図 2: Random CSP

5.3 他手法との比較実験

本論文で提案する手法の有効性を検証するために，有名な CSP の近似解法である GENET[Davenport 1994] と LPPH-CSP2 との比較実験を行った．GENET は各変数が制約違反を最小にするように値を変更することを繰り返す．もし，局所解に陥ってしまったら，現在違反してる制約の重要度をあげてやることにより局所解からの脱出を図る．N-Queen 問題，Car Sequencing 問題を各手法で解いた結果を図に示す．図 3，4 の縦軸は解の探索に要した平均 CPU 時間，横軸は各手法で解いた問題を示している．図 3,4 の実験結果より問題に依存するが，LPPH-CSP2 と GENET は同程度の効率であることがわかる．

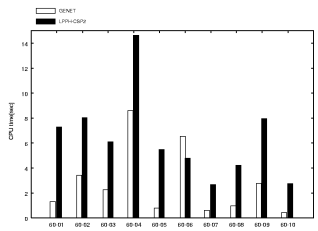

図 3: Car Sequencing 問題

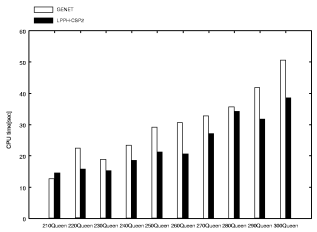

図 4: N-Queen 問題

6. まとめ

GENET は変数を同期して更新すると解の振動現象を起こしてしまうため解を得ることができなくなってしまう．このため GENET では非同期で変数を更新していく必要がある．一方，本論文の提案手法では全ての変数を同期して解を求める．このため，提案手法の方が GENET よりもハードウェア化した場合，高速に解をもとめることができると考えられる．

本論文で提案した手法を最大 CSP といった CSP の拡張問題に対して適用させることが今後の課題である．また，3 節に示した制約以外の組み合わせ最適化問題で一般的に現れる制約についても提案手法に組み込むことも必要である．

参考文献

[Davenport 1994] A.Davenport, E.Tsang, C.j.Wang, and K.Zhu: GENET: A Connectionist Architecture for Solving Constraint Satisfaction Problems by Iterative Improvement, National Conference on Artificial Intelligence, pp.325-330, 1994.

[Nagamatu 1995] M.Nagamatu and T.Yanaru: On the stability of Lagrange programming neural networks for satisfiability problems of propositional calculus, Neurocomputing, 13, pp.119-133, 1995.