2003－MPS－47　(18)
2003／12／12

最小支配集合問題の近似解抽出アルゴリズム

森山 隆人[1]　富田 悦次[1]

概要　無向グラフ中の最小支配集合を 1 つ求める問題は NP 困難のクラスに属している．最小支配集合は最小の節点数で全節点に隣接するという点に着目して，欲張り法を基礎とした確率的探索アルゴリズムを提唱する．本稿では，アルゴリズムを実働化し，ランダムグラフと DIMACS の提供するグラフに対して計算機実験を行い，アルゴリズムの有効性を示す．

An Approximation Algorithm for the Minimum Dominating Set Probrem

Takato MORIYAMA[1] and Etsuji TOMITA[1]

Abstract　The problem of finding a minimum dominating set is known to be NP-hard. A minimum dominating set has a few vertices which are adjacent to all vertices. Then we propose a new randomized algorithm based on a greedy algorithm. We show here its effectiveness by computational experiments for some random graphs and DIMACS graphs.

1　はじめに

組合せ最適化問題は多くの重要な問題を含むが，それらの大部分は NP 困難のクラスに属し，多項式時間で厳密解を求めるアルゴリズムは存在しがたい．そのため，短時間で結果を得るには近似解法を用いることになるが，近似解の解精度を上げつつ，実行時間が短いアルゴリズムであることが重要である．

本稿では最小支配集合問題 (Minimum Dominating Set Problem) を対象とし，効率的な解の抽出方法を用いた近似アルゴリズムを提案し，この問題に対して有効であることを実験的に示す．

2　表記法と定義

本稿で対象とするのは節点集合 V，枝集合 E から成る無向グラフ $G=(V,E)$ であり，その節点には生成過程などからある順序付けがなされているものとする．なお，以下では無向グラフ $G=(V,E)$ を単にグラフと呼ぶこととする．

2 節点 $u,v \in V$ を両端点とする枝を，u と v の非順序対 $(u,v)=(v,u) \in E$ で表し，この時 u と v は隣接しているという．また，節点 u に隣接している節点集合を隣接節点集合と呼び，$\Gamma(u)$ で表す．なお, $\Gamma(u)$ の要素数を次数と呼び，$|\Gamma(u)|$ で表す．

グラフ $G=(V,E)$ の節点部分集合 $V' \subseteq V$ が支配集合であるとは，$V-V'$ に対して少なくとも 1 つの節点 $v' \in V'$ が隣接していることをいう．このとき V' は V 中の全ての節点を支配するという．

また最小支配集合問題とは与えられたグラフ G に対して要素数が最小の支配集合を求めることである．

3　アルゴリズム nDS

本稿で提唱する確率的アルゴリズム nDS を図 1 に示し，以下で説明を行う．

[1]電気通信大学大学院 電気通信学研究科 電子情報学専攻
Graduate School of Electro-Communications,
The University of Electro-Communications

```
procedure nDS (G = (V, E),T,max_iter)
begin
  V' := ∅ ;
  DS := V ;
  max:=0
  for t := 1 to max_iter · |V| do
    while |V' ∪ Γ(V')| ≠ |V| do
      for i := 1 to |V| do
        if |Γ(v_i)| ≥ max then
          if sigm(−ΔEnerg(v_i)/T) > rand[0, 1) then
            x_i:=v_i ;
            max:=|Γ(v_i)|
          fi
          未支配の節点に繋がる枝を残し,
          x_i が支配している節点への枝を
          次の選択候補から外す
        fi
      od
    od
    if |V'| < |DS| then
      DS := V
    fi
  od
end
```

図 1: アルゴリズム nDS

ここでグラフ $G = (V, E)$ において，$V = \{v_1, v_2, ..., v_i, ..., v_{|V|}\}$ とする．また V の部分集合 V' は $\{x_1, x_2, ..., x_i, ..., x_{|V|}\}$ の各節点要素からなる．

3.1 解候補集合の導出方法

アルゴリズムの基本は次数の最も大きい節点を解候補集合に優先して採用していくことにより，解候補集合を導出する方法による．最小支配集合は最小の節点数で全節点に隣接しているので，最小支配集合に含まれる節点は次数が高くなることが多いためである．

節点を採用していく際に既に支配されている節点への枝次数を考慮するのは無駄であるので，解候補集合に節点が採用された時点で，枝を削除して全節点の枝次数を更新し，再度枝次数の高い節点を選択して，全ての節点が支配されるまで，これを繰り返す．

枝を削除する際に，vertex cover 問題における欲張り法 [4] のように解候補に選択された節点により支配された節点への枝を全て削除してしまうと問題が生じる場合がある．ここで vertex cover 問題とはグラフ $G = (V, E)$ 中の全ての $(u, v) \in E$ に対し，$u \in V'$ または $v \in V'$ が成り立つような $V' \subseteq V$ を求める問題である．

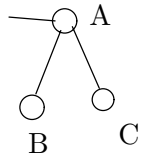

図 2: ある部分グラフ

例えば図 2 のようなグラフ上の節点において，節点 A が支配され，かつ節点 B,C が支配されていない場合，節点 A に隣接する全ての節点を削除してしまうと，節点 A を解候補集合に選択すれば節点 B,C を 1 つの節点で支配できるのに，節点 B,C をそれぞれ選ばなくてはならず，最終的に解候補集合の要素数が増えてしまうことがある．そこで，最小支配集合問題では解候補集合の節点によって支配されていない節点への枝は残して，枝を削除するようにする．

3.2 節点の受理関数

次数が同じ節点が複数存在する場合や最高次数の節点が最小支配集合に選択されない場合を考慮し，適切な受理関数を用いて節点の取捨選択を行う．受理関数によって 1 回ごとに探索される解候補集合に差異を持たせ，これを複数回実行することにより解精度の向上を図る．

節点 v_i に対する受理関数には，節点の次数に重きを置く関数を用いる．アルゴリズム nDS では，アニーリング法 [3] に用いられているシグモイド関数を基本とした関数を用いている ((2) 式)．シグモイド関数は 0 から 1 までの実数を出力する連続関数であり，代入するエネルギー関数がある程度の値以上になるとほぼ 1 となるため，高い次数の節点を優先的に受理する場合に適している．そこで，このシグモイド関数を着

表 1: ランダムグラフに対する実行結果（解精度・時間）

Graph			RaDS[2]		nDS0		nDS	
n	p	DS		[sec]		[sec]		[sec]
100	0.2	7	8.30	(4.153)	8.00	(0.007)	**7.66**	(0.802)
	0.3	5-6	6.00	(2.980)	6.00	(0.000)	**5.33**	(0.574)
	0.4	4	4.60	(2.503)	4.66	(0.000)	**4.00**	(0.464)
	0.5	3-4	**3.53**	(1.955)	4.00	(0.003)	3.77	(0.458)
	0.6	3	3.00	(1.585)	3.00	(0.000)	3.00	(0.340)
	0.7	2-3	2.87	(1.332)	2.67	(0.000)	**2.67**	(0.327)
	0.8	2	2.00	(1.117)	2.00	(0.000)	2.00	(0.241)
	0.9	2	2.00	(0.883)	2.00	(0.000)	2.00	(0.249)
200	0.25	7-8	8.83	(31.44)	8.00	(0.010)	**7.37**	(6.096)
	0.4	5	5.77	(21.49)	5.33	(0.000)	**5.00**	(4.418)
	0.5	4	4.30	(17.05)	4.67	(0.000)	**4.00**	(3.668)
	0.6	3	3.50	(13.86)	3.67	(0.000)	**3.00**	(2.880)
	0.8	2	2.00	(9.135)	2.00	(0.000)	2.00	(1.935)
300	0.25		9.97	(114.0)	9.00	(0.013)	**8.03**	(23.95)
	0.4	5	6.03	(75.85)	6.00	(0.013)	**5.63**	(18.33)
	0.5	4	5.00	(60.69)	4.67	(0.010)	**4.07**	(15.42)
	0.6	4	4.00	(49.46)	4.00	(0.007)	4.00	(13.63)
	0.75	3	3.00	(36.61)	3.00	(0.007)	3.00	(11.32)
500	0.25		11.80	(570.6)	10.00	(0.053)	**9.33**	(142.0)
	0.5	5	5.67	(300.6)	5.00	(0.030)	**5.00**	(87.12)
	0.6	4	4.23	(244.5)	4.00	(0.020)	**4.00**	(72.43)
	0.7	3	3.67	(197.5)	3.67	(0.013)	**3.00**	(59.48)
	0.8	3	3.00	(157.6)	3.00	(0.017)	3.00	(57.23)
	0.9	2	2.00	(121.4)	2.00	(0.010)	2.00	(42.01)

目節点の解候補集合中の節点として選択する確率として採用する．

エネルギー関数は節点の次数を用いて以下のように定める．

$$\Delta Energ(v_i) \stackrel{def}{=} \frac{|V| - |\{v_i\} \cup \Gamma(v_i)|}{|V|} \quad (1)$$

節点 v_i の受理関数は，このエネルギー関数のシグモイド関数の値とする．ここで T は固定温度パラメータである．

$$sigm \quad (-\frac{\Delta Energ(v_i)}{T}) = \quad (1 + \exp(\frac{\Delta Energ(v_i)}{T}))^{-1} \quad (2)$$

4 計算機実験

アルゴリズムを C 言語でプログラム作成し，実働化した．また確率アルゴリズム RaDS[2] と，グラフの厳密解を求めるために厳密解抽出アルゴリズム DSC[1] を，出来る限り同条件になるように共通化してプログラム作成し，それぞれに同一のグラフデータを入力して実行することにより比較実験を行った．実行時間はグラフデータの読み込みなどを含まないアルゴリズム本体のみの処理時間とした．これらの実行に使用した計算機環境は CPU：Intel Pentium4 3.0GHz, OS：Linux である．各アルゴリズムに対するパ

ラメータ設定は表 2 のとおりとした．

表 2: 各パラメータ設定

Algorithm	T	max_iter
RaDS	0.15	10
nDS	0.2	10

まず，ランダムグラフに対して各々のアルゴリズムを適用した結果の一部を表 1 に示す．表において n は節点数，p は枝の存在確率，DS は DSC により導出された最小支配集合の要素数の範囲である．グラフは n と p の各組合せごとに，乱数のシードを変えたものを 3 個用意した．提示してあるのは 3 個のグラフに対して得られた結果の平均値である．nDS の受理関数を用いずに，最高次数の節点が見つかった時点で解候補集合に採用するアルゴリズムを nDS0 と名付ける．この実行結果を比較のため掲載する．この結果から受理関数による解精度の向上が確認できる．また，nDS が RaDS に比べ全体的に解精度，実行時間において優れていることが分かる．その上，受理関数による解精度の向上が無くても RaDS よりも良い結果が出ている箇所があり，次数に着目した探索が最小支配集合問題に対して有効であることが分かる．

次に，DIMACS で提供されているクリーク問題のベンチマークテスト用グラフに適用した結果を表 3 に示す．これらのグラフは枝の張り方が偏っている，もしくは全く均一である様な特殊なグラフである．今回は p_hat300-1 のグラフの実行結果において RaDS に比べ nDS が解精度，実行時間において優れており，また，他のグラフにおいても短時間で同等の解精度を得ることが出来ることが確認できた．

5 むすび

ランダムグラフや DIMACS のグラフを対象とした計算機実験により，最小支配集合問題に対する近似解抽出アルゴリズムとして，提唱したアルゴリズム nDS が有効であることが確認できた．

今後の課題として，受理関数を変更するなどのアルゴリズムの変更による改良，遺伝的アルゴリズム (GA) 等，他のアルゴリズムとの比較，及び融合手法の開発が挙げられる．

表 3: DIMACS グラフに対する実行結果（解精度・時間 [sec]）

Graph	DS	RaDS[2]	nDS
p_hat300-1	-	8.9(123.2)	**7.0(20.64)**
p_hat300-2	-	4.0(62.31)	4.0(13.41)
p_hat300-3	-	3.0(37.32)	3.0(11.20)
johnson8-2-4	3	3.0(0.032)	3.0(0.010)
johnson8-4-4	2	2.0(0.365)	2.0(0.100)
johnson16-2-4	3	3.0(1.970)	3.0(0.771)
johnson32-2-4	3	3.0(116.8)	3.0(63.09)
hamming6-2	2	2.0(0.248)	2.0(0.072)
hamming6-4	4	4.0(0.475)	4.0(0.110)
hamming8-2	2	2.0(12.43)	2.0(5.172)
hamming8-4	2	2.0(17.13)	2.0(5.675)
c-fat200-1	-	13.0(27.36)	13.0(10.17)
c-fat200-2	-	6.0(15.04)	6.0(5.230)
c-fat200-5	-	3.0(7.406)	3.0(2.580)
c-fat500-5	-	6.0(215.7)	6.0(95.22)
san200_0.7_1	2	2.0(10.84)	2.0(2.151)
san200_0.7_2	3	3.0(11.47)	3.0(2.813)
san400_0.7_1	3	3.0(92.11)	3.0(29.45)
san400_0.7_2	3	3.0(93.80)	3.0(29.48)
san400_0.7_3	3	3.0(94.73)	3.0(29.37)
san400_0.5_1	-	4.0(111.1)	4.0(37.05)

【謝辞】 検討いただいた若月光夫助手に感謝します．また，本研究は電通大研究・教育活性化支援システムの支援を受けている．

参考文献

[1] 梅田徳之, 富田悦次, 三木哲也, “ Dominating Set 問題に対する分枝限定アルゴリズムとその実験的評価,” 人工知能全大, pp.243-255 (1998).

[2] 梅田徳之, 富田悦次, 三木哲也, “確率アルゴリズムに基づいた支配集合問題の近似解法,” 情報処理学会研究報告, 2000-MPS-29-1, pp.1-4 (2000).

[3] Emile Aarts and Jan Korst,“Simulated Annealing and Boltzman Machines,” John Wiley & Sons (1989).

[4] V.V. ヴァジラーニ,“近似アルゴリズム,”シュプリンガー・フェアラーク東京株式会社 (2002).