A-017

整数計画法によるグラフ焼失数の厳密計算

An integer programming approach for computing the burning number of a graph

宮岡 誠† 鮭川 矩義‡ 浅野 孝夫†

Miyaoka Makoto Sukegawa Noriyoshi Asano Takao

1. はじめに

ソーシャルグラフなどにおける影響伝播のしやすさを測る指標にグラフ焼失数（burning number）がある．これは，隣接する頂点間で火が燃え移る（＝影響が伝播する）と考え，グラフを焼き尽くす（＝影響が全体に行き渡る）のに要する時間として定義される．焼失数の計算は一般に NP-困難であるが，パスやサイクルなどの特殊なグラフに対しては焼失数が簡単に計算できることが知られている．しかし，一般のグラフに対して焼失数を計算する方法はなく，焼失数の挙動については不明な部分が多い．

本研究では，一般のグラフに対して焼失数を厳密に計算する整数計画問題による定式化を提案する．その有効性を数値実験によって検証し，さらに，計算時間の短縮方法について議論する．

2. 問題

2.1 グラフ焼失数

無向グラフGを考える．頂点集合をV，枝集合をEと書き，$n = |V|$とする．また，離散時刻$T = \{1,2,\dots\}$を考える．焼失数の計算において，各頂点は「未燃焼」と「燃焼中」のいずれかの状態をとる．時刻$t = 1$では，すべての頂点が未燃焼とする．ここで，以下のルールを考える：

Step 0.	$t = 1$とし，すべての頂点を未燃焼とする
Step 1.	すべての頂点が燃焼中ならば終了
Step 2.	未燃焼の頂点があれば，そのうちの 1 つを選び，燃焼中にする（着火する）
Step 3.	すべての未燃焼の頂点iに対し，iに隣接する燃焼中の頂点があれば，iを燃焼中にする
Step 4.	$t = t + 1$とし，Step 1 へ

グラフ焼失数とは，このルールに従って燃焼中の頂点を増やしていくとき，すべての頂点が燃焼中になるまでにかかる時間（つまり，Step 1 で終了したときの，tの値）の最小値として定義され，$b(G)$と表記される．

つまり，燃焼数が小さければ，影響が広まりやすいと言える．たとえば，完全グラフでは焼失数は2，つまり，$b(K_n) = 2$と計算される．図 1 に，より一般的なグラフに対する例を示す．ここで，未燃焼の頂点は「白」，燃焼中の頂点は「黄色」，各時刻で着火された頂点は「赤色」で示されている．この結果では，$t = 3$として Step 1 に入ったときに全ての頂点が燃焼中であることがわかり，計算が終了する．$t = 2$で終了することはできないので，このグラフの燃焼数は，3と結論付けられる．また，図 1 の例では，

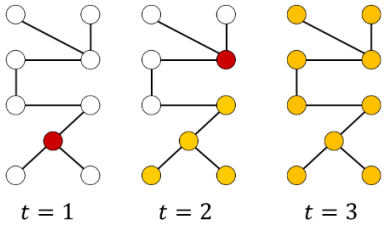

図 1　グラフ燃焼の様子

他にもさまざまな最適解（$t = 3$で終了できるような頂点の着火順）が存在することがわかる．

2.2 先行研究

グラフ焼失数は，Bonato, Janssen, Roshanbin [1] によって提唱された比較的新しい影響伝播モデルである．基本的な影響伝播モデルでは，最初の意思決定で最終的な結果が決まる場合が多い．これに対し，グラフ焼失数では，どの順に頂点を着火するかという多段階の意思決定を考えており，この点が問題を複雑にする．実際，同じ著者ら[2]によって，グラフ焼失数の計算が NP-困難であることが示された．一方，パスやサイクルなどの特殊なグラフに対しては解析が進んでいる．特に，パスの焼失数は$\lceil\sqrt{n}\rceil$であるのだが，一般の上界$2\lceil\sqrt{n}\rceil - 1$があり，これが$\lceil\sqrt{n}\rceil$にまで下げられる（つまり，パスがタイトな例である）というのが主要な未解決の予想である．より洗練された上界については Bessy ら[3]を参照されたい．

本研究では，先行研究のような極値的な調査ではなく，グラフ焼失数の一般的な性質を調べることを目的とし，整数計画法に基づく厳密かつ効率的なグラフ焼失数の計算方法の確立を目指す．

3. 定式化

3.1 整数計画問題

本研究では，グラフ焼失数問題を，以下の 0-1 整数線形計画問題(IP)として定式化することを提案する：

$$
\begin{aligned}
\min\quad & b && && \\
\text{s. t.}\quad & \textstyle\sum_{i\in V} p_{it} \le 1 && (t \in T) && (1)\\
& x_{it} \le p_{it} + \textstyle\sum_{\{i,j\}\in E} x_{j,t-1} && (i \in V, t \in T\backslash\{1\}) && (2)\\
& x_{i,1} = p_{i,1} && (i \in V) && (3)\\
& s_t \ge (n - \textstyle\sum_{i\in V} x_{it})/n && (t \in T) && (4)\\
& b = \textstyle\sum_{t\in T} s_t && && (5)
\end{aligned}
$$

† 中央大学　Chuo University
‡ 東京理科大学　Tokyo University of Science

変数x_{it}は，頂点iが時刻tに燃焼中であれば1，未燃焼であれば0を取る．変数p_{it}は，時刻tで頂点iに着火するとき1，そうでなければ0を取る．また，変数s_tは時刻tにおいて全ての頂点が燃焼中であれば0，そうでなければ1を取る．

式(1)は，各時刻で着火できる頂点は高々ひとつであることを要請する．式(2)は，ある頂点iに着目し，時刻$t-1$において，iに隣接する頂点の中に燃焼中のものがあるか，あるいは，時刻tでiが着火されたときには，時刻tでiが燃焼中であることを（目的関数の最小化に基づいて）要請する．式(3)は，時刻1において，着火された頂点を除き，すべての頂点が未燃焼であることを要請する．式(4)は，時刻tにおいて，未燃焼の頂点がひとつでもあれば，s_tが1を取ることを要請している．したがって，s_tの総和の最小化が目的であり，これは式(5)で表される．

3.2 計算時間の短縮

予備実験から，3.1 節の(IP)では（高性能な最適化ソルバーを使っても）解けない問題例がいくつもあることが確認された．そこで，ここでは，先ほどの(IP)に対し，制約式の追加および目的関数の修正を行い，計算時間の短縮を目指す．まず，以下の制約を考える：

$$x_{it} \geq x_{i,t-1} \qquad (i \in V, t \in T\backslash\{1\}) \qquad (6)$$

$$x_{it} \geq x_{j,t-1} \qquad (\{i,j\} \in E, t \in T\backslash\{1\}) \qquad (7)$$

式(6)は，各頂点に対し，時刻$t-1$で燃焼中であれば，時刻tでも燃焼中であることを要請する．これは定式化としては冗長であるが，探索空間を限定できるため，ソルバーで繰り返し解かれる(IP)の線形緩和問題などで有効にはたらく可能性がある．式(7)も同様である．

また，図 1 の例からもわかるように，グラフ焼失数問題には，一般に，さまざまな最適解がある．この性質は，多くの最適化ソルバーで採用されている分枝限定法（最適解のある空間を見極め，探索空間を限定していく方法）との相性が極めて悪い．そこで，最適解に「順序」をつけるべく，以下の修正を提案する：

$$c = \sum_{i\in V, t\in T} x_{it} \qquad (8)$$

$$b = \sum_{t\in T} s_t + 1 - c/K \qquad (9)'$$

ここで，式(8)で定義された非負整数変数cは「早い段階でグラフの大部分が燃焼中になる」度合を定量化したものとみなせる．言い換えると，次数の高い頂点や，隣接する頂点の次数が高い頂点などへの着火が優先されるようなモデルである．定数Kはc/Kが 1 未満になるような大きな正の整数である．実際には，Kはn^2にすれば十分である．この修正により，bの値が同じ場合，cの値が大きいものがより優れているとみなされ，最適解に順序がつく．

4. 数値実験

(IP)および追加の制約式(6), (7)の有効性を検証する．最適化ソルバーには Gurobi 7.0.2 を用いた．また，計算環境は，CPU は Intel Core i7-6700(3.4GHz)，メモリ容量は 32.0GB である．問題例は，ソーシャルグラフのベンチマーク問題集[4]から 5 つ選んだ．表 1 に結果を示す．表 1

表 1　定式化による計算時間 (秒) の違い

n	(IP)	(IP)+(6)	(IP)+(7)	(IP)+(6),(7)
34	0.02	0.03	0.06	0.08
77	0.08	0.03	0.31	0.31
112	0.09	0.09	0.63	0.70
332	1.95	1.42	26.64	26.33
512	>7,200.00	767.57	>7,200.00	6,802.98

の結果から，式(6)が有効であることがわかる．特に，頂点数 512 の時には大幅な改善がみられる．実際，(IP)がこの 512 頂点の問題例を厳密に解き終えるには，24,468 秒，つまり，7 時間弱の時間がかかった．つまり，約 97%の時間短縮を達成した．以下，その理由を考察する．ある頂点iが時刻tで初めて燃えたとする．このとき，0-1変数$x_{i,t'}$（t'はt以上の任意の整数）は焼失数に対応する時刻t^*に1になっていればよいので，時刻tからt^*までの間，0と1のどちらの値をとっても，目的関数値に影響しない．式(6)を追加することで，これらの変数の値が一意に定まり，計算時間が短縮されたと考えられる．また，上で言及した変数のように値が一意に定まらない変数の個数は，対象とするグラフの焼失数が大きいほど，多い．そのため，式(6)は，焼失数の大きい問題例に対して有効であることが予想される．実際，実験に使った頂点数 332 以下の 4 つのグラフは，焼失数が 2 か 3 であったのに対し，頂点数 512 のグラフは焼失数が 6 であった．

また，表 1 には載せていないが，式(8),(9)による目的関数の修正も効果が見られた．たとえば，頂点数 512 の問題例に対し，計算時間は 3567.88 秒にまで短縮された．しかし，これらが一般的に有効であるか，あるいはどういった制約式と目的関数の組合せが良いかについては，検討の余地がある．より詳細な結果については当日報告する．

5. まとめと今後の課題

本研究では影響伝搬モデルにおける基本的な指標であるグラフ焼失数に着目し，その整数最適化問題としての定式化を与えた．試した問題例はまだ少ないものの，頂点数 500 程度の比較的大きなグラフに対しても効率よく焼失数を計算できることが確認できた．

一方，より良い定式化の探究や，最終的な目標である，焼失数と関係の深いグラフパラメータの探求は，今後の課題である．また，応用場面を意識して，近似解法を構築することも重要であると考える．この手始めとして，提案した整数最適化問題(IP)とその線形緩和問題(LP)の目的関数値の乖離（整数性ギャップ）の解析を考えている．

参考文献

[1] Bonato A, Janssen J, and Roshanbin E. How to burn a graph. Internet Mathematics. 2016 Mar 3; 12 (1-2): 85-100.

[2] Bonato A, Janssen J, and Roshanbin E. Burning a graph is hard. arXiv preprint arXiv:1511.06774. 2015 Nov 20.

[3] Bessy S, Bonato A, Janssen J, Rautenbach D, and Roshanbin E. Bounds on the burning number. Discrete Applied Mathematics. 2018 Jan 30; 235: 16-22.

[4] http://www-personal.umich.edu/~mejn/netdata/