線形な領域計算量の最良優先探索の探索順序の緩和*

4J－6

大下和宏　栗原正仁　大内東†

北海道大学工学部‡

1 はじめに

人工知能における問題空間の探索法の分野で、限られたメモリーの上で実行できる探索法について、これまでに様々な方法が考え出されている。中でもR.E.Korfが示した再帰的最良優先探索法 (Recursive Best-first Search 以下 RBFS) は、領域計算量が線形のオーダーですみ、かつ節点の探索順序は最良優先探索と同じく、最小のコストを持つ節点を優先して展開する探索法である。この探索法の特徴は時間計算量は発生した節点の数に比例することですが、問題を解く途中で同じ節点を再び展開しなければならないことがあり、それによって節点の重複が起こり、そのため発生する節点の総量は従来の最良優先探索よりも相対的に多くなり、問題を解く速さに影響する。そこで本研究では、速さを高めるために、開節点の中での最小のコストにある程度近い値を持つ節点であれば探索を打ち切らずに展開を行うようにして、発生する節点の総量を押さえる探索法を考え、RBFSとの計算の速さ、解の質などについて比較実験を行っている。本稿ではその概要を示す。

2 Recursive Best-First Search

2.1 RBFSのアルゴリズム

RBFSは名前の通り再帰的に呼び出される手続きで、節点を展開するごとに呼び出される。RBFSは、しきい値を越えたコストを持つ節点を切り落として探索していくが、そのしきい値は各呼び出しの際に局所的に与えられる値である。各呼び出しの際のしきい値は、展開する節点の兄弟にあたる節点の中での最小のコストと、親の節点の際の呼び出しでのしきい値との、どちらか小さい方とする。

各節点についてコストとして保持される値があり、それを $F(n)$ と表し、また評価関数によって求められる値を $f(n)$ と表すと定義する。ここでRBFSは3引数の関数で $RBFS(n, F(n), b)$ と表す。(b はしきい値)

RBFSのアルゴリズムは以下の通りである。

$RBFS(n, F(n), b)$

1: 節点 n が終点なら、アルゴリズム終了

2: 節点 n を展開して、子節点がなければ
無限大を返す

3: 各子節点に対して $F(n_i)$ の値を求める
そのとき、もし $f(n) < F(n)$ ならば
$F(n_i) \leftarrow max(F(n), f(n_i))$
そうでなければ $F(n_i) \leftarrow f(n_i)$

4: 子節点の中で $F(n_i)$ が最小のものを n_1,
その次に小さいものを n_2 とする

4.1: $F(n_1) > b$ ならば、$F(n_1)$ を返す

4.2: $F(n_1) \leftarrow RBFS(n_1, F(n_1), min(F(n_2), b))$
として4に戻る

RBFSの第二引数 $F(n)$ は、節点 N が過去に一度展開されたことがある場合、親節点の持つ値が子節点のコストより大きければ、その値を子に継承させることで効率を上げるために用意されたものである。

RBFSの最初の呼び出しの際の各引数は、根節点 r、$f(r)$、しきい値は無限大をあたえる。

2.2 RBFSに関する定理

RBFSに関する定理を以下に示す。

・定理1:

$RBFS(n, F(n), b)$ が実行されるときは必ず $F(n) \leq b$ である。

・定理2:

$b \neq \infty$ のとき、$RBFS(n, F(n), b)$ は節点 n 以下を、開節点の中でコストが b を越えないものがある限り展開を続け、その中で終点が見つからなければ節点 n 以下の部分木の中の開節点の内の最小のコストの値を返す。

*Relaxing best-first order in linear-space best-first search

†Kazuhiro OHSHITA, Masahito KURIHARA, Azuma OHUCHI

‡Faculty of Engineering, Hokkaido University

・定理3:
$RBFS(n, F(n), b)$ が呼び出される条件は、$F(n)$ は節点 n の子孫にあたるもの中の開節点のコストの最小値以下で、かつ b は n の子孫でない開節点のコストの最小値以下である。
・定理4:
節点がRBFSによって展開されるときは、その節点のコストはその時点でのどの開節点のコストよりも小さいか同じ値である。

これらの定理より $RBFS(r, f(r), \infty)$ は最良優先の順に探索を行うことが証明される。

2.3 RBFSの計算量

ルートから現在探索している節点までの各節点でRBFSの再帰呼び出しが行われ、各地点の兄弟の節点が記録されているので、領域計算量は $O(bd)$ となる。(b は展開の際に発生する子節点の数の平均値で、d は探索される深さの最大値)

RBFSの時間計算量は発生した節点の総数に比例した値となる。

2.4 RBFSの問題点

定理2より $RBFS(n, F(n), b)$ は節点 n 以下でしきい値 b を越えないコストの節点を展開するが、その中で終点がなければ開節点のコストの最小値を返すだけで、発生した節点は記憶されずに、節点 n の他の兄弟節点の展開を行う。そのため、他の兄弟節点以下の探索が終点が見つからずに終り、新たなしきい値で再び n 以下の探索を行う際、前回展開した節点についても再び展開しなければならない。こうしたことで発生する節点の総量が多くなり、時間計算量にも影響される。

3 RBFSの探索順序の緩和

発生する節点の累計を押さえるために、RBFSが呼び出される際のしきい値の与え方を変更する。しきい値を与える際にその値をある程度増加させることで、探索順序は最良優先ではなくなるが、本来のしきい値で打ち切らずに探索を進める。これによって同じ節点に対して展開を繰り返す回数が押さえられると期待できる。

しきい値の増加方法の一つとして、他の兄弟の節点のコストの最小値にある定数 c を足したものと、前の呼び出しの際のしきい値との小さい方を選ぶ方法を考えた。具体的にはRBFSのアルゴリズムのステップ4.2で新たに $F(n_1)$ に与える値を

$RBFS(n_1, F(n_1), min(F(n_2) + c, b))$

に変えるのみである。

もう一つの方法として、ある定数を足すのではなく、定数 a を掛ける、すなわちRBFSの変更箇所が

$RBFS(n_1, F(n_1), min(F(n_2) \times a, b))$

となる。この方法では、終点に近づくにつれ、しきい値は小さくなるため増加量も小さく、探索順序が最良優先に近くなると予想されます。

両方とも変更箇所がしきい値の与え方のみなので、領域計算量はRBFSと変わらず $O(bd)$ のままである。

これらの方法で予想されることは、コストがある程度近ければ深い方の節点を展開するので、与える定数が大きくなるにつれ、解の長さ(終点までのステップ数)が大きくなることである。また定数が余りに大きくても、探索順序が深さ優先に近づくために節点の総量の減少は起こらないと考えられる。

4 比較実験

実験に使う問題として、15パズルを使用する。評価関数 $f(n)$ は

$f(n) = g(n) + 3h(n)$

で表される。ここで $g(n)$ は節点 n の根からの深さ、$h(n)$ は全てのタイルの終点までのマンハッタン距離の合計である。しきい値をかえるための定数の値を複数選び、様々な条件のもとで問題を解き、RBFSとの節点の総量、速さ、解の長さなどについて比較を行う。

5 おわりに

本稿ではRBFSから節点の総量をを押さえるための緩和法を示す。これからはこの方法の実際の効果を調べ、また最良優先の探索順序を緩和することによって、解に起こる影響を調べる。これらの実験結果については、当日に発表する。

参考文献

[1] R. E. Korf, Linear-space best-first search, *Artificial Intelligence* 62 (1993) 41-78.

[2] R. E. Korf, Depth-First Iterative-Deepening: An Optimal Admissible Tree Search, *Artificial Intelligence* 27(1) (1985) 97-109.