A-009

上書きハッシュ表の性質

Advantage of Overwrite Hash Table

†山口 陽平 Yohei Yamaguchi
†岩田 茂樹 Shigeki Iwata

1 はじめに

上書きハッシュ表は、データを書き込む際にハッシュ値で与えられたスロットに常に書き込む方式のハッシュ表である。そのスロット上にすでに別のデータがあった場合は、そのデータは上書きされて削除される。この研究では、上書きハッシュ表をゲーム木の探索に利用した場合、どの程度計算時間を削減するかを解明する。

従来のチェーン方式やオープンアドレス方式のハッシュ表 [2] では、書き込むデータ量はハッシュ表の大きさを超えられない制限がある。またデータ量が多くなるとデータの衝突が発生してハッシュ表の性能を低下させる。上書きハッシュ表では、書き込むデータ量に制限はなくデータの衝突も発生しない。一方で元々保持していたデータが上書きされることにより失われることがある。

2 上書きハッシュ表の利用法

ゲーム木の探索を考える。ある局面に対して、同一の局面の結果をすでに探索していてその結果が保存されていれば、それを利用することでその局面を再度探索することを省くことができる。上書きハッシュ表を使ってデータを保存し、計算時間の短縮をはかる。

保存した探索結果を利用する前に同じハッシュ値を示す他のデータによって上書きされ利用できなくなる確率は、同一局面が現れるまでのハッシュ表への書き込みの回数に依存する。探索する順番の中で同一局面が近くに集中して現れれば、利用したいデータが上書きによって失われる確率が低くなると考えられる。そして、いくつかの種類のゲームは実際に同一局面が近くに集中すると考えられる。

3 上書きハッシュ表の計算時間の短縮

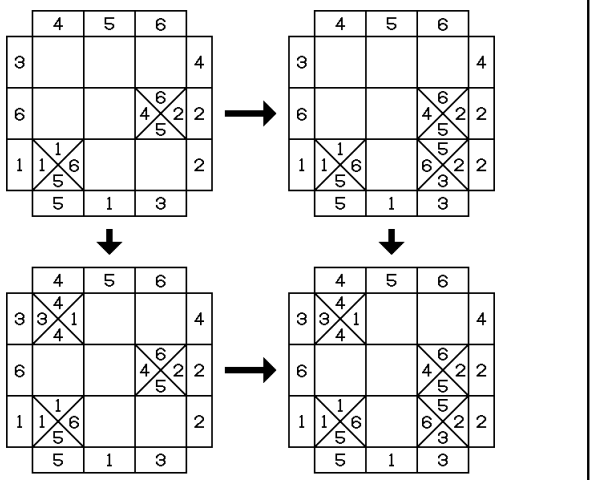
いくつかの仮定のもと、上書きハッシュ表がどの程度計算時間短縮するかを数理的に計算することを試みる。ハッシュ表に用いるハッシュ関数は、出力であるハッシュ値が一様に分布する一様分布であると仮定する。扱うゲームは、盤面に駒を置いていく形式の一人用パズルとする。ゲーム木上のある特定の深さ（すでに置かれた駒の数と等しい）でのみ上書きハッシュ表を利用するものとする。同一局面は同じ深さにしか現れないためである。この深さを ℓ とする。深さ ℓ 現れた頂点について、それがこの深さにおいて何番目に現れたものなのかを、「時刻」と呼ぶことにする。

上書きハッシュ表の計算時間短縮効果を計算するためには、同一の局面が現れるタイミングと、その際に前に保存したデータが上書きハッシュ表に残っていて計算量の削減に成功する確率を考慮する必要がある。

同一の局面が現れるタイミングについて考える。盤面に駒を置いていく形式の一人用パズルゲームでは、駒を置く順番だけを入れ替えても現れる局面は同一である。i 手目においてどのマスにどの駒を置くかという選択を s_i とおけば、ℓ 手目の局面はこれまで選択の列 $< s_1, s_2, \ldots, s_\ell >$ で表せる。この時、$< s_1, s_2, \ldots, s_\ell >$ の中の s_i の任意の並び替えも同一の局面を表す。よって駒を置いていく形式のゲーム木の探索では、ℓ 手目の局面と同一の局面は最大で $\ell!$ 回現れる。しかし実際には、考えられる全てのゲーム展開をチェックする前に探索が終了したり、探索の工夫によって一部のゲーム展開のチェックを省くことが考えられるため、同一局面を各 $\ell!$ 回ずつ探索することは無いと考えられる。

ここでは、3 手前まで同じ選択をし、2 手前の選択と 1 手前の選択の順番のみが違う 2 つの同一局面の位置関係のみを考え、それ以上遠い祖先を共有する局面は考えない。ゲーム木で言えば、2 世代前の祖先を共有するいとこの関係である。図 1 は、タイル張りパズル (Tiling puzzle) において、同じ局面に至る例を表している。図 2 は深さ $\ell - 2$ の祖先を共有する同一局面のゲーム木上での位置関係を表している。

図 1. 違う手順で同じ局面に至る例



共有する祖先は深さ $\ell - 2$ の頂点の中では k 番目の頂点であるとし、2 手前と 1 手前の選択をそれぞれ i, j とする。計算を簡略化するために、2 手前と 1 手前において選択の数は共に等しく d とし、選択 i は i 番目に現れるものとする。深さ ℓ の頂点を根とする各部分木を探索した際に正解に辿り着かず計算が終了しない確率を p とする。深さ $\ell - 2$ の頂点の総数は $d^{\ell-2}$ とする。このとき、同一局面が現れる時刻の差は

$$(jd + i) - (id + j)$$

と表わせる。この間に上書きハッシュ表上のデータが削除されずに残っている確率は、ハッシュ表のスロットの数を M と

† 電気通信大学大学院 情報理工学研究科 情報・通信工学専攻

し、$\alpha = 1 - 1/M$ とすると

$$\alpha^{(jd+i)-(id+j)-1}$$

と表せる。これに 2 回目の局面が現れるまで計算が続く確率を考慮した上で、k, i, j $(0 \leq k < d^{\ell-2}, 0 \leq i < j < d)$ についての総和を取れば、2 世代前の祖先を共有する深さ ℓ のノード全体でハッシュ表からデータを得られる回数の期待値 E_ℓ が得られる。すなわち

$$E_\ell = \sum_{k=0}^{d^{\ell-2}-1} p^{kd^2} \sum_{j=1}^{d-1} \sum_{i=0}^{j-1} p^{jd+i} \alpha^{(jd+i)-(id+j)-1}$$

である。上書きハッシュ表による計算時間の削減を考えない場合の深さ ℓ の頂点の総数 T_ℓ は

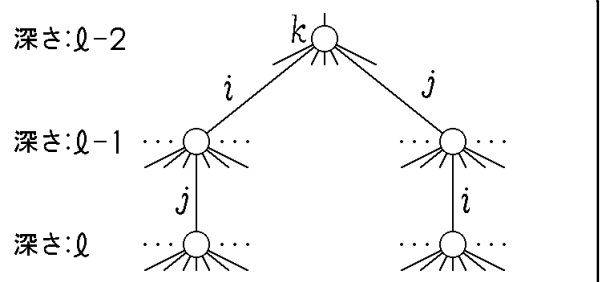
$$T_\ell = \sum_{k=0}^{d^{\ell-2}-1} p^{kd^2} \sum_{j=1}^{d-1} \sum_{i=0}^{j-1} p^{jd+i}$$

であるため、深さ ℓ 以下の部分木の探索時間について、上書きハッシュ表を利用する場合と利用しない場合の比は

$$1 - \frac{E_\ell}{T_\ell} = 1 - \frac{p^d \alpha^{d-1} \dfrac{1-(p^d\alpha^{d-1})^{d-1}}{1-p^d a^{d-1}} - p^{d+1} \dfrac{1-(p^{d+1})^{d-1}}{1-p^{d+1}}}{\alpha \dfrac{1-p^{d^2}}{1-p} (1 - p\alpha^{1-d})}$$

と表せる。

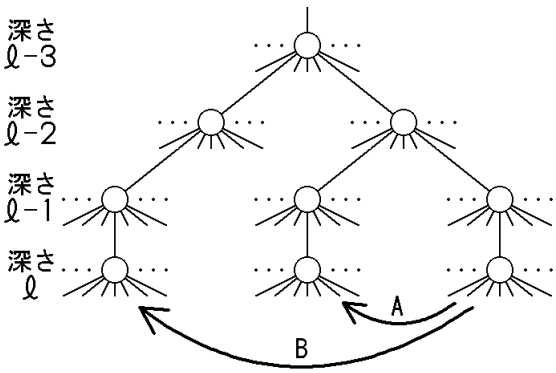
図 2. 2 世代前の祖先を共有する同一局面の例



例として $d = 30, M = 100000, p = 0.99$ として代入すると、値は 0.89 となり、約 11% 計算時間が短縮される。

実際には 2 世代前の祖先を共有する同一局面に限らず、3 世代前、4 世代前、…、を共有する同一局面に関しても上書きハッシュ表に残したデータの再利用は起こる可能性があり、それによって計算時間はさらに減る。しかし、さかのぼる祖先が遠くなるほど、それを共有する頂点の数は指数関数によって増えていくのに対し同一局面の数は階乗関数によるため、同一局面の密度が下がり、その時刻差の距離は大きくなる。これが、2 節で述べた同一局面が近くに集中する性質である。2 世代前の祖先を共有する同一局面（図 3 の A）において局面データが上書きされずに保たれる確率より、3 世代以上前で枝分かれした同一局面（図 3 の B）において局面データが上書きされずに保たれる確率の方が小さいため、計算時間の削減の回数の期待値も無視できるほどに小さいものと考えられる。

図 3. 共有する祖先と同一局面間の距離の関係



ただしこの計算は、前述したような、探索の工夫によって一部のゲーム展開のチェックを省いたり、パズルの解に至りやすい選択肢から優先的に選択して p が変化するなどの可能性を無視している。

ハッシュ表のスロット数を M とし、初めにすべてのスロットは空白とする。時刻 t $(t > 0,\ 0 \leq k \leq M)$ の後で上書きハッシュ表がちょうど M スロットを使用する確率を $P(t,k,M)$ とする。仮定より $P(0,0,M) = 1$, $P(t,0,M) = 0$ $(t \leq 1)$, $P(t,k,M) = 0$ $(0 \leq t < k)$ である。$t \geq 0$, $1 \leq k \leq M$ とする。$P(t+1,k,M) = (k/M)P(t,k,M) + \{(M-k+1)/M\}P(t,k-1,M)$ が成立する。$S(t,k) = \{M^t(M-k)!/M!\}P(t,k,M)$ とおくと $S(t,k)$ は**第 2 種のスターリング数** (例えば [1] を見よ) であり、$S(t,k)$ が解ける。すなわち

$$P(t,k,M) = \frac{1}{M^t} \binom{M}{k} \sum_{i=1}^{k-1} (-1)^i \binom{k}{i} (k-1)^t$$

となる。時刻 t に M スロット中に存在する局面数は $\sum_{k=1}^{M} kP(t,k,M)$ なので、これを変形して $M(1-\alpha^t)$ を得る。これにより上書きハッシュ表でのヒット確率が求まる。[3]

4 上書きハッシュ表による計算時間短縮効果の測定実験

実際にパズルを解くプログラムを作成し、上書きハッシュ表の効果を測定した。題材としてタイル張りパズルを取り上げる。これは、駒の四方の色を隣接する駒や枠と一致させて配置するパズルである。盤面の大きさや色の数やハッシュ表を用いるレベルなど条件をいろいろ変えて、盤面をランダムに生成した問題で試してみたところ、深さ ℓ 以下の部分木の探索時間について、上書きハッシュ表を利用する場合と利用しない場合の比はほぼ 0.8 から 1.0 の間であった。

参考文献

[1] Knuth: The Art of Computer Programming Vol.1, Fundamental Algorithms, 3rd Ed., Addison-Wesley (1997)
[2] Knuth: The Art of Computer Programming Vol.3, Sorting and Searching, 2nd Ed., Addison-Wesley (1998)
[3] 山口, 岩田: 上書きハッシュ表の性質, 電子情報通信学会 2013 年総合大会学生ポスターセッション ISS-P-231(2013)