1L-01

確率微分方程式の統計量計算へのフロンティア法の利用と工夫

濱　晴矢†　　大久保 潤‡

埼玉大学工学部情報工学科†,‡

1. はじめに

世の中の様々な現象は，時間とともに偶発的に変化する要因を考慮した確率微分方程式として記述され得る．確率微分方程式の時間発展を考えることは有益であり，モンテカルロ法によるシミュレーションが利用されることが多い．しかし，複雑な系や長時間発展においてその計算量は膨大であり，計算上の工夫が必要となる．その工夫の一つに，確率過程の双対性を利用して，確率微分方程式の統計量を離散的な過程を通じて計算する枠組みがある[1, 2]．特に最近の研究により，双対過程から導出される状態遷移の directed walk を重み付きで計算することにより，組合せ論的に統計量を計算できることがわかっている[3]．

先行研究[3]では動的計画法での計算がなされている．しかし，次元数や状態遷移の幅が大きくなると計算量爆発が生じてしまう．そのため，重みの影響が少ない経路を無視するなど，近似的な計算が重要となる．その近似的な計算の検討の際に経路毎の重みの情報は有用であるが，動的計画法ではその情報を得ることができない．

本研究では経路毎の重みを計算するための効率的な列挙アルゴリズムを提案する．パラメータ変更後の再計算などを念頭に置き，列挙には木構造を用いる．ただし，素朴に構造を保持すると，ステップ数の増加により計算量だけでなく必要メモリも膨大となる．また，木構造の枝刈りアルゴリズムであるフロンティア法を素朴に用いるだけではメモリ圧縮率が低い．そのため，提案手法では到達経路のみの探索と分割数による分類といった工夫を行うことにより，圧縮効率を高める．

2. 組合せ論における問題設定

本研究では確率微分方程式の例として noisy van del Pol 方程式を用いる．この方程式は 2 変数系で，振動現象を示すことが知られており，データ駆動アプローチの研究でも利用されている[4]．Noisy van del Pol 方程式の場合，2 次元格子上における 10 種類の状態遷移が双対過程から導出される[3]．このとき，任意の始点からM回の移動で終点(原点)に到達する directed walk を考える．なお，この組合せ論的な問題設定において，始点が計算したい統計量に対応すること，負の座標への遷移を無視してよいことなど，いくつかの特徴がわかっている．

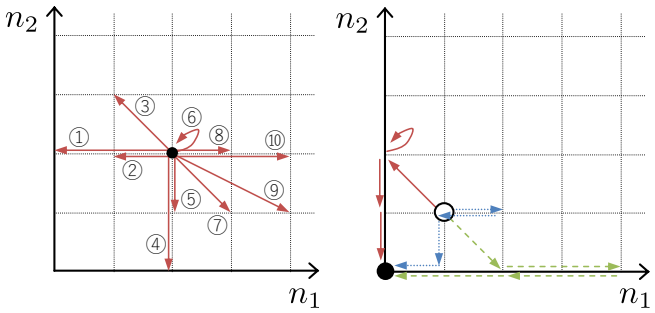

図 1：Noisy van del Pol 方程式から導出される状態遷移(左)と Directed walk の例(右)．

3. 統計量計算の提案手法

提案手法は，大きく分けて次の 3 つの手順からなる．

・到達する組合せの探索
・分割数による分類
・フロンティア法による木構造の構築

3.1 到達する組合せの探索

上述したように，directed walk においては負の座標に遷移する経路を無視できる．そのため状態遷移の組合せにおいて，遷移の順番を考慮する必要がある．これは，もし遷移の順番により負の座標領域に到達してしまう walk があれば，無視してよいからである．一方で，順番まで考慮すると組合せの数が多くなってしまう．

そこで提案手法では，最初に状態遷移の組合せを考え，到達するかどうかをのみを探索する．その後，フロンティア法[5]を用いることで，構築する木構造において順番を考慮して計算を実施し，全体の計算量を減らす狙いがある．

例として，状態遷移数 3 種類，ステップ数 3 回の場合を考えると，順列では $3^3 = 27$ 通りの組合せが出てくる．一方，順番を考慮しない組合せは ${}_{3+3-1}C_3 = 10$ 通りで済む．

Application of Frontier-based methods for computing statistics of stochastic differential equations
† Hareruya Hama, Saitama University
‡ Jun Ohkubo, Saitama University

3.2 分割数による分類

分割数とは，ある自然数を 0 以上の自然数の和で表す考え方である[6]．ここでは分割数を，あるステップ数において，状態遷移を区別せずに各状態遷移が何回発生するかに対応させる．

例えば状態遷移数 3 種類，ステップ数 3 回の場合，組合せでは(1,1,1)，(1,1,2)，(1,1,3)，(1,2,2)，(1,2,3)，(1,3,3)，(2,2,2)，(2,2,3)，(2,3,3)，(3,3,3)と表せる．一方，分割数による分類では，(A,A,A)，(A,A,B)，(A,B,C)と表すことで，同じとみなせる組合せをまとめることができる．

この分割数による分類は，前節で導入した組合せの考え方に利用可能である．結果として，メモリやファイルサイズの圧縮が可能となる．

3.3 フロンティア法による木構造の構築

フロンティア法は，グラフの部分構造列挙の結果を表す ZDD の構築アルゴリズムである[5]．図 2 のように，同じ状態とみなせるノードを共有することで枝刈りを行う．なお，図 2 において，ノードの(3.0.0)の数字は分割数による分類における(A,A,A)を表し，どの状態遷移が何回残っているかを示している．木構造はトップダウン型の構築を採用しているため，必要メモリも削減される．

4. 実験方法

確率微分方程式の統計量計算を想定し，他手法との木構造のサイズと計算量の比較を目的とした数値実験を実施する．実験には noisy van del Pol 方程式から導出される 10 種類の状態遷移を用いる．そして，ステップ数毎の必要メモリと計算時間を計測する．

木構造のサイズについては，①全列挙，②分割数を用いない枝刈りのみのフロンティア法，および③提案手法を比較して，メモリの圧縮率を考察する．また計算量については，先行研究である動的計画法との計算時間および計算量を比較し，考察する．

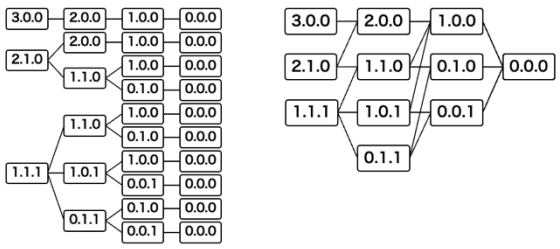

図 2：分割数を用いて全列挙した場合（左）と
フロンティア法で枝刈りした場合（右）．

表 1 ：各手法における必要メモリ（byte）

ステップ数	①	②	③
1	-	-	-
2	600	24	332
3	8,000	216	538
4	100,000	2,528	1,850
5	1,200,000	24,000	3,847
6	14,000,000	232,200	8,379
7	160,000,000	2,145,612	16,766
8	1,800,000,000	19,711,104	33,770
9	20,000,000,000	178,654,248	67,418
10	220,000,000,000	1,610,159,400	138,463

5. 実験結果および考察

表 1 に木構造を保存するために必要なメモリサイズを示す．ステップ数を増やすにつれて考慮すべき状態数が増えるため，必要メモリが増加する．表 1 より，工夫なしのフロンティア法②と比べ，分割数による分類を用いた提案手法③の圧縮効率が高いことがわかる．ただし，ステップ数が低い場合には，③に必要な保存メモリが②と比べて大きくなっている．これは到達する組合せと木構造を分けて保存していることが要因の一つとして挙げられる．

計算量については，提案手法よりも動的計画法の方が高速であった．動的計画法は，次元数と状態遷移の幅が計算量に関わるため，今回の 2 次元系かつ状態遷移の幅が -2 から +2 と小さい場合では非常に高速となる．なお，高次元系の場合には次元数に左右されない提案手法が適している可能性もある．また，枝刈りを見据えた列挙の用途には動的計画法を利用できないことから，今後，提案手法の活用を計画している．

謝辞 本研究は JSPS 科研費 JP21H05843 の助成を受けている。

参考文献

[1] C. Giardinà, et al., “Duality and hidden symmetries in Interacting particle systems,” J. Stat. Phys. 135, 25 (2009).
[2] J. Ohkubo and Y. Arai, “Duality in stochastic processesfrom the viewpoint of basis expansions,” J. Stat. Mech. 2019, 063202 (2019).
[3] J. Ohkubo, “Combinatorics for calculating expectation values of functions in systems with evolution governed by stochastic differential equations,” J. Stat. Mech. 2021, 013401 (2021).
[4] N. Črnjarić-Žic, et al., S. Maćešić and I. Mezić, “Koopman operator spectrum for random dynamical systems,” J. Nonlinear Sci. 30, 2007 (2020).
[5] S. Minato, “Techniques of BDD/ZDD: Brief history and recent activity,” IEICE Transactions on Information and Systems E96.D, 1419 (2013).
[6] G.E. Andrews and K. Eriksson, Integer Partitions (Cambridge Univ. Press, 2004).