2V-04

成果蓄積進化型知識発見手法における突然変異率制御の検討

前田　一樹†　　嶋田　香‡
群馬大学†　　群馬大学‡

1. はじめに

成果蓄積進化型知識発見手法として GNP (Genetic Network Programing)ベースの手法である GNMiner が存在している[1].この手法では, ルール発見数が一定数を超えるまでかかる時間が全体の中で大きな割合を占めるという課題が存在している.これを短縮するために突然変異率の制御をする研究は, これまで行われていなかった.

時点毎でのルール発見数から求められる指標に応じて変異率を調整することで, 全体的な世代数を短縮し発見の効率化が可能なのではないかと考えた.本研究では過程としてのルール発見数に応じてステージ分けし, それに応じて進化操作における変異率を変化させることで改善するという方法を検討した.

結果として, 変異率を調整することによって必要な平均世代数が減少した.

2. GNMiner について

GNMiner とはグラフネットワークの個体を複数作成して成果蓄積型の集団進化によりユーザにより事前に設定された興味深さ条件を満たすアイテム集合やアソシエーションルール群を発見する方法である.GNMiner はグラフネットワークを用いてルールを発見する.GNMiner で用いられるグラフの例を図 1 に示す.また, 用いられるデータの形を表 1 に示す. X,Y は目的属性である. 本研究では, ルールの前件部を属性の組み合わせとし, 結論部を連続値をとる属性の統計的な分布に関する条件を満たす場合を扱っている. GNMiner のノードには二種類あり, 処理ノード(P)と判定ノード(J)が存在している.
それぞれのデータに対して, 最初にP_1ノードから出発し, 矢印の通りの遷移先に遷移しつつ, 判定ノードの属性がデータにおいて 1 であった場合,

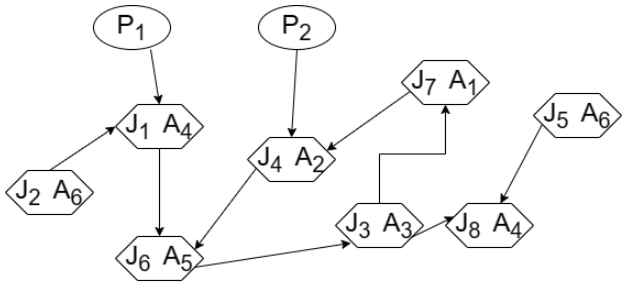

図1:GNMinerの構造

表 1:データの例

属性	A_1	A_2	A_3	A_4	A_5	A_6	X	Y
データ 1	1	0	0	1	1	1	12.3	2.6
データ 2	0	0	1	1	1	0	5,6	-28

その属性をアソシエーションルールの前件部を満たすものとして次に遷移する. 規定数のノードを遷移したら終了し, 次の処理ノードP_2から開始する. このグラフを一個体として個体を複数生成し, 計算を行う.

全ての処理ノードからの処理が終わった後, 進化を行う. 各個体は遺伝的アルゴリズムの手法である選択, 変異, 交叉によって進化させていく.

GNMiner の変異は二種類あり, 一つは判定ノードの遷移先が変異するもの(変異 1), 過去 5 世代分で発見されたルールに応じて判定ノードが担当する属性が変異するもの(変異 2)である. 進化が終わったら次の世代に進み進化させた個体で同じことを行う.

個体からアソシエーションルールが発見されたとき, 新規ルールならば保存する. 新規ルールであるかどうかの確認として, ルールプールにそれまで発見されたルールを保存しておき, ルールプール中の各ルールと比較を行う. 同じものがなければ新規ルールと判断する.
ルール発見数が一定数に達するか, 世代数が一定の値に達した時点で 1 ラウンドとする. ラウンドが切り替わるたびにすべての個体とルールプールは初期化する.

3. 関連研究

先行研究として[2]がある. ここでは進化操作に

Controlling Mutation Rates on Outcome-accumulating Evolutionary Rule Discovery Method
† Kazuki Maeda, Gunma University
‡ Kaoru Shimada, Gunma University

おける変異率は世代を通して一定値としたときにおけるルール発見数に応じたStage1からStage3に状態の分類，評価手法としてQI(Query Efficiency Index)を考案し計測している.QIとは，現在のラウンドでのルール発見数と新規ルール確認のためのルールプール累計比較回数を用いて計算したものである.計算式は現在のルール発見数をm,累計比較回数をnとして$QI=2m/(n(n-1))$と計算される.この値は全く無駄がない理想的な状態においては1となる.

現在の課題として，GNMinerではルール発見数がある程度大きくなるまでの時間が長いことがあった.先述の研究ではいくつかの変異率を用いた実験を行っているものの，一つのラウンドの中で変異率を変えることは行っていない.本研究では成果蓄積型の手法を扱うため，従来のエリート個体の獲得を目指す遺伝的アルゴリズムの利用における変異率の操作法とは目的が異なっている.ラウンド中にステージ数に応じて変異率を変えることを提案するために，ラウンド中での突然変異率制御のための検討が課題となる.

4. 実験

ルール発見数1～20を第一ステージ，21～200を第二ステージ,201～2000を第三ステージとして,変異確率を変更する.

変えるパラメータとしては変異1，変異2，交叉の確率である.変異率を変える方法としては以下の表2のようにした.表2の中の数字はそれぞれ(変異1，変異2，交叉)の確率である

表2:パラメータ設定

	ステージ1	ステージ2	ステージ3
Change	1/4,1/4,1/5	1/6,1/6,1/10	1/10,1/10,1/20
Change_inv	1/10,1/10,1/20	1/6,1/6,1/10	1/4,1/4,1/5
High	1/10,1/10,1/20	1/10,1/10,1/20	1/10,1/10,1/20
middle	1/6,1/6,1/10	1/6,1/6,1/10	1/6,1/6,1/10
Low	1/4,1/4,1/5	1/4,1/4,1/5	1/4,1/4,1/5

すべて表のとおりに変えたもの(all)，交叉のみ変えたもの(kousa)，変異1だけ変えたもの(muj)，変異2だけ変えたもの(mua)，それらのみ変えなかったもの(allkousa，allmuj，allmua)といった場合についても実験をした.変えないパラメータは最も低い値(1/4,1/4,1/5)を用いた.

5. 結果と今後の課題

評価として各ステージの平均世代数と

表3:実験の結果

	1st_stage	2nd_stage	3rd_stage	Sum	Drop
change_all	67.25833	55.19583	28.98646	151.4406	0.040
change_all-kousa	66.53403	65.19267	26.31518	158.0419	0.045
change_all-mua	67.86906	61.38437	16.17951	145.4329	0.053
change_all-muj	66.30681	63.49005	24.71728	154.5141	0.045
change_kousa	67.58745	72.66358	15.81070	156.0617	0.028
change_mua	67.50315	76.40861	25.21324	169.1250	0.048
change_muj	67.42360	76.52055	16.16017	160.1043	0.051
change_inv_all	54.32797	29.29260	19.63558	103.2562	0.067
change_inv_all-kousa	55.18335	26.15174	19.08957	100.4247	0.051
change_inv_all-mua	54.55876	30.36966	29.59081	114.5192	0.064
change_inv_all-muj	53.79167	26.57906	20.86538	101.2361	0.064
change_inv_kousa	55.07314	28.00109	32.97598	116.0502	0.084
change_inv_mua	55.29558	26.36354	22.10895	103.7681	0.073
change_inv_muj	55.26811	28.83568	30.69838	114.8022	0.075
high_all	54.29533	29.17698	33.67210	117.1444	0.079
high_all-kousa	62.78396	41.34652	31.20214	135.3326	0.065
high_all-mua	57.25343	28.04329	17.79725	103.0940	0.053
high_all-muj	57.73044	33.52960	29.45983	120.7199	0.054
high_kousa	60.82347	42.16511	15.40187	118.3904	0.037
high_mua	64.73441	56.17742	27.03871	147.9505	0.070
high_muj	65.56408	55.95693	16.73319	138.2542	0.048
middle_all	63.06263	43.76964	22.05839	128.8907	0.058
middle_all-kousa	65.93803	55.48504	20.55128	141.9744	0.064
middle_all-mua	64.56029	49.80042	15.87214	130.2328	0.038
middle_all-muj	64.43908	52.88971	20.18592	137.5147	0.048
middle_kousa	64.78170	61.18607	15.10083	141.0686	0.038
middle_mua	65.29343	70.97564	20.06780	156.3369	0.056
middle_muj	66.86432	68.91667	15.90812	151.6891	0.064
low_all	66.68004	87.78247	16.50581	170.9683	0.053

ルール発見数が一定数に達しなかった脱落の割合を用いる.

表3はステージごとの平均世代数，合計(Sum),脱落割合(Drop)の表である．今回の結果で，ラウンド中に変異率を変えることによって,変えないときよりも良い効果が出たと考える.事前の予想では変異率を先に小さくし，のちのステージでは大きくする変化方法が最も良い結果となった.

把握と性能の評価の検討を進めることにより,変異率制御法の開発が課題となる.計算効率監視等を目的とした新指標を開発しそれを用いることでのデータに自己適応する方法を考えたい.

参考文献

[1] K. Shimada, T. Arahira, S. Matsuno, ItemSB: Itemsets with Statistically Distinctive Backgrounds Discovered by Evolutionary Method, International Journal of Semantic Computing, 16 (3), 2022, 357-378

[2] S. Matsuno, K. Shimada, Evolutionary Operation Setting for Outcome Accumulation Type Evolutionary Rule Discovery Method Proc. of the Genetic and Evolutionary Computation Conference 2022, 451-4542