Vol.2021-BIO-65 No.4
2021/3/11

Elementary flux mode 型代謝経路の化学量論的な代謝ネットワーク構造の算出: 算出されたネットワーク構造間の異同判定のためのアルゴリズムの最適化

太田潤[†1,a)]

概要：代謝ネットワークにおける“経路”として，化学量論的に釣り合った個々の反応の集合としての“経路”を考えることができる．この反応の集合としての“経路”を elementary flux mode 型経路（EFM 型経路）と呼んでいる．特定の EFM 型経路を定義する，経路に含まれる反応に関する情報（どの反応がいくつ含まれるか）から，その EFM 型経路の化学量論的に釣り合った代謝産物レベルネットワーク構造（化学量論的な代謝ネットワーク構造）を求めることは，1 つの数理的な問題ととらえられる．化学量論的な代謝ネットワーク構造は，経路に含まれる反応がどのような順序とつながりで働いて経路の原料分子群から目的分子群が生成するかを示す. EFM 型経路の化学量論的な代謝ネットワーク構造の算出においては，算出される経路のネットワーク構造の異同判定が全体の計算速度に影響を与え得る．前稿では，ネットワーク構造を，ネットワーク構造内の経路をノードとするネットワーク構造とみなして異同判定を行うとともに，2 つの異同判定されるネットワーク構造の構造全体の間でのつながりの一致確認の前に，部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返しを行うことにより，異同判定アルゴリズムの高速化を試みた．本稿では，ネットワーク構造内の経路をノードとするネットワーク構造の異同判定，部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返しに，Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値, EC 値）の利用を加えた 3 者の組み合わせによる，異同判定アルゴリズムの最適化を試みた．

キーワード：代謝ネットワーク，elementary flux mode，グラフ同型判定

Calculation of stoichiometric metabolic network structure of elementary flux mode type metabolic pathway: Optimization of algorithm for difference judgement between calculated network structures

JUN OHTA[†1,a)]

1. はじめに

代謝ネットワークの主要な機能の 1 つである生理活性物質生成のための“経路”に関する知識・情報は，医学・生物学的に重要である．代謝ネットワークにおける“経路”（代謝経路）に，化学量論的に釣り合った個々の反応の集合としての“経路”[1,2] がある．その代表例は，elementary flux mode［1］と extreme pathway［2］であり，本稿ではこれらの反応の集合としての“経路”を elementary flux mode 型経路（EFM 型経路）と呼ぶ.

EFM 型経路においては，経路を定義する反応の集合に含まれる反応の情報から経路の収支式が求まり，収支式左辺に現れる原料分子群のすべてが揃った場合に収支式右辺の目的分子群が生成すると言えるが，反応の集合自体は，原料分子群から目的分子群に至る具体的な道筋（EFM 型経路に含まれる反応がどのような順序で働くか）を示さない.

EFM 型経路に原子レベルの情報を与える（原子レベルマッピングを行う）とき，その副産物として経路の化学量論的な（化学量論的に釣り合った）代謝産物レベルの代謝ネットワーク構造が求まることを私は経験した［3,4,5］．多くの一般的な代謝マップが化学量論を考慮していない（解糖系では化学量論的に 1 分子の glucose から 2 分子の lactate が生成するが代謝マップ上には glucose ノードと lactate ノードがそれぞれ 1 つしかない等）が，EFM 型経路の原子レベルマッピングの副産物として求まる代謝産物レベルの代謝ネットワーク構造は，経路の化学量論を正確に表現するもので，原料分子群から目的分子群に至る具体的な道筋（EFM 型経路に含まれる反応がどのような順序で働くか）を示していた．また，1 つの EFM 型経路に対して求まる化学量論的な代謝産物レベルネットワーク構造は 1 種類とは限らなかった［3,4］.

これまでの代謝産物レベルの生合成代謝経路に関する情報学的研究には，目的代謝産物の生成に必要な原料代謝産物の集合を求める手法の研究［6］や与えられた代謝ネットワークに存在する生合成経路の列挙を試みる研究［7］などがあるが，前者の手法では生合成経路の算出はできず，

†1 岡山大学　大学院医歯薬学総合研究科（医）生化学分野
a) jo25@md.okayama-u.ac.jp

後者の手法で算出される生合成経路は，化学量論を考慮しない 1 代謝産物（種）1 ノードの代謝産物レベルネットワーク構造である．

EFM 型経路の原子レベルマッピングの副産物として経路の代謝産物レベルの化学量論的な代謝ネットワーク構造が求まったが，原理的に，代謝産物レベルのネットワーク構造は原子レベル情報に依存しない．そして，特定の EFM 型経路を定義する，経路に含まれる反応に関する情報（どの反応がいくつ含まれるか）から，その EFM 型経路の化学量論的な代謝産物レベルネットワーク構造を求めることは，1 つの数理的な問題ととらえられる．私は，この問題に取り組み，原子レベルの情報を用いることなく，EFM 型経路を定義する代謝産物レベルの反応情報のみから EFM 型経路の化学量論的な代謝産物レベルネットワーク構造を算出するためのアルゴリズムを報告した［8,9］.

報告した［8,9］のアルゴリズムには，異同判定アルゴリズムの段階が含まれる．ネットワーク構造が複数算出された場合に,それらの異同判定を行う必要があるからである．複数回現れる代謝産物が多い EFM 型経路の場合，算出される構造が多いため，異同判定アルゴリズムが，化学量論的な代謝産物レベルネットワーク構造算出アルゴリズム全体の計算速度に影響を与える．そこで［8,9］で用いた異同判定アルゴリズムの高速化を試みてきた. 前稿［10］では，ネットワーク構造を，ネットワーク構造内の経路をノードとするネットワーク構造とみなして異同判定を行うとともに，2 つの異同判定されるネットワーク構造の構造全体の間でのつながりの一致確認の前に，部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返しを行うことにより，異同判定アルゴリズムの高速化を試みた．本稿では，ネットワーク構造内の経路をノードとするネットワーク構造の異同判定，部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返しに，Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値, EC 値）［11,12］の利用を加えた 3 者の組み合わせによる，異同判定アルゴリズムの最適化を試みた．

本稿で扱う“化学量論的な代謝ネットワーク構造”は，代謝産物と反応をノードとするネットワーク構造である．代謝産物に対応するノードを“代謝産物ノード”，反応に対応するノードを“反応ノード”と呼ぶ．2.1.1 に述べる変換の後の EFM 型経路においては，すべての代謝産物ノードそれぞれが例外なく 2 つの反応ノード（その代謝産物ノードに対応する代謝産物分子を生成する反応の反応ノードと利用する反応の反応ノード）のみとつながりを持つ.

EFM 型経路の化学量論的な代謝ネットワーク構造は，グラフとみなすことができる．EFM 型経路の化学量論的な代謝ネットワーク構造をグラフ $G = (V, E, L, \ell)$とみなす場合，V は代謝産物ノードと反応ノードに対応する頂点の集合, E は代謝産物ノードから反応ノードに至るエッジまたは反応ノードから代謝産物ノードに至るエッジに対応する辺の集合，L は代謝産物の種類または反応の種類に対応する（頂点の）属性の集合，ℓ は頂点である代謝産物ノードにはその属性としての代謝産物の種類を，頂点である反応ノードにはその属性としての反応の種類を対応付ける関数である．このように考えると, EFM 型経路の化学量論的な代謝ネットワーク構造の異同判定は，グラフ同型判定であり，グラフ同型判定アルゴリズムにより異同判定がなされ得る．

本稿の異同判定（グラフ同型判定）アルゴリズムは，“部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返し”による異同判定を行う木構造探索を含むが,“段階的に部分構造の範囲を拡げながらの繰り返し”の際に，属性が同じ頂点すべてからなる部分構造を一度に範囲に加えて構造の適合性を確認する点で［13,14］と異なる．また，あらかじめ定めた木構造において探索する点で，木構造を生成しながら探索する［14］と異なる．

2. 方法

EFM 型経路の化学量論的な代謝ネットワーク構造算出の概要，グラフ同型判定アルゴリズムの構成要素，グラフ同型判定のためのコアアルゴリズム，グラフのグラフ内経路を頂点とするグラフへの変換，グラフの頂点の属性への拡張隣接関係値の追加，グラフ同型判定アルゴリズム *GCore*・${}_{p}$*GCore*・${}_{+EC}$*GCore*・${}_{+EC}{}_{p}$*GCore*，EFM 型経路の化学量論的な代謝ネットワーク構造算出アルゴリズムとグラフ同型判定アルゴリズムの一体化について述べる. 本稿では，配列 Z の j 番目の要素を“$Z[j]$”と表す記法を用いている．

2.1 EFM 型経路の化学量論的な代謝ネットワーク構造算出の概要［8,9］

経路を構成する反応の集合として与えられた EFM 型経路の化学量論的な代謝ネットワーク構造算出の概要を述べる．

2.1.1 与えられた EFM 型経路の，単一の仮想原料反応から単一の仮想目的反応に至る EFM 型経路への変換［8,9］

反応の集合として与えられた EFM 型経路全体の収支から，経路の化学量論的な原料代謝産物 source(s)（原料分子群）と目的代謝産物 target(s)（目的分子群）を求める．原料代謝産物と目的代謝産物は何れも複数個，複数種の分子である場合がある．この原料代謝産物と目的代謝産物に対して，原料代謝産物を何もないところから生成する仮想原料反応 source reaction と目的代謝産物を除去するが何も生成しない仮想目的反応 target reaction を定義する．

source reaction: no metabolite(s) → source(s)
target reaction: target(s) → no metabolite(s)

この 2 つの仮想反応を反応の集合に追加すると，与えられ

た EFM 型経路は，単一の仮想原料反応から単一の仮想目的反応への経路に変換される．この変換により，原料代謝産物と目的代謝産物の代謝産物ノードを含むすべての代謝産物ノードそれぞれが例外なく 2 つの反応ノード（その代謝産物ノードに対応する分子を生成する反応の反応ノードと利用する反応の反応ノード）のみとつながりを持つ EFM 型経路のネットワーク構造を考えることができるようになる．2.1.2 と 2.1.3 における“EFM 型経路”は変換後の EFM 型経路である．

2.1.2 EFM 型経路の反応構成の反応番号による記述および反応番号と代謝産物番号による反応の記述［8,9］

化学量論的な代謝ネットワーク構造の算出のために，EFM 型経路に現れる反応，代謝産物に対して，反応番号，代謝産物番号を付す．ある反応・代謝産物が経路に複数回現れる場合，それらに同じ反応番号･代謝産物番号を付す．

EFM 型経路の反応構成を反応番号の配列（反応番号配列）として表現する．EFM 型経路に同じ反応番号の反応が複数回現れる場合は，それらを別々の要素として格納する．

EFM 型経路に現れる反応はそれぞれ，反応番号，基質の代謝産物番号の配列，生成物の代謝産物番号の配列の組み合わせとして表現する．ある反応の反応式の基質側の辺に代謝産物番号が m で係数が n の代謝産物分子があれば，その反応の基質の代謝産物番号の配列に m が n 回現れるようにする．生成物についても同様である．仮想原料反応 source reaction の基質の代謝産物番号の配列，仮想目的反応 target reaction の生成物の代謝産物番号の配列は空配列である．

2.1.3 EFM 型経路の化学量論的な代謝ネットワーク構造の算出［8,9］

EFM 型経路の化学量論的な代謝ネットワーク構造に存在する反応ノードは，EFM 型経路の反応番号配列の要素と 1 対 1 に対応する．各反応ノードは，その反応の基質に相当するすべての代謝産物ノードそれぞれからのエッジが入る地点であるとともに，その反応の生成物に相当するすべての代謝産物ノードそれぞれに向けてのエッジが出る地点でもある．すなわち，各反応ノードには，代謝産物（基質）が入るエッジとそのエッジを介して入る代謝産物（基質）の組み合わせの集合と，代謝産物（生成物）が出るエッジとそのエッジを介して出る代謝産物（生成物）の組み合わせの集合が対応する．EFM 型経路の化学量論的なネットワーク構造に含まれるすべての代謝産物ノードはそれぞれが，1 エッジを介して 1 反応ノードから出て，別の 1 エッジを介して別の 1 反応ノードに入る．したがって，EFM 型経路の化学量論的なネットワーク構造において，エッジを介して反応ノードから出る代謝産物ノードの集合を P，エッジを介して反応ノードに入る代謝産物ノードの集合を S とすると，集合 P と集合 S は代謝産物ノードの集合としては同一の集合でなければならない．すなわち，集合 P の要素（どの反応ノードから出るかにより区別される代謝産物ノード）は集合 S の要素（どの反応ノードに入るかにより区別される代謝産物ノード）と同一ノードとして 1 対 1 に対応付けられる．P の特定の要素と S の特定の要素は，両方が同じ（種類の）代謝産物のノードである場合に限り対応付けできる（“対応付け条件”）．化学量論的な代謝ネットワーク構造はこの対応付けにより生成する．“対応付け条件”を満たす，P の要素と S の要素の対応付けを 1 つ求め x とすると，P の要素と S の要素の可能な対応付けは，すべてが，x により対応付けられた要素（P の要素に対応付けられた S の要素，S の要素に対応付けられた P の要素の何れか）の置換により得られる．

EFM 型経路の化学量論的な代謝ネットワーク構造算出においては，この対応付け（置換）に対応して生成し得るすべてのネットワーク構造から unique なものを見出すことを行う．そのためのアルゴリズムとして，アルゴリズム B＋付加ステップを用いる［9］．このアルゴリズム（アルゴリズム B＋付加ステップ）では，“対応付け条件”を満たす置換を互換の繰り返しとして行い，互換によりネットワーク構造が算出されるごとに異同判定アルゴリズムでネットワーク構造の異同判定を行う．異同判定アルゴリズムは，以下に述べるグラフ同型判定アルゴリズムを用いる．

2.2 グラフ同型判定アルゴリズムの構成要素

概要および 1 に“本稿では，ネットワーク構造内の経路をノードとするネットワーク構造の異同判定，部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返しに，Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値，EC 値）の利用を加えた 3 者の組み合わせによる，異同判定アルゴリズムの最適化を試みた”と述べた．

グラフ同型判定アルゴリズムは，“グラフ同型判定のためのコアアルゴリズム”，“グラフのグラフ内経路を頂点とするグラフへの変換”，“Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値，EC 値）の利用”の組み合わせからなる．“部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返し”による異同判定に“グラフ同型判定のためのコアアルゴリズム”が，“ネットワーク構造内の経路をノードとするネットワーク構造”に“グラフのグラフ内経路を頂点とするグラフへの変換”が，“Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値，EC 値）の利用”に“グラフの頂点の属性への拡張隣接関係値の追加”が対応または関係する．

2.3 グラフ同型判定のためのコアアルゴリズム

本項の“グラフ同型判定のためのコアアルゴリズム”は，“部分ネットワーク構造レベルでのつながりの一致確認の，段階的に部分構造の範囲を拡げながらの繰り返し”による異同判定に対応する．このアルゴリズムは，単独で使用される以外に，“ネットワーク構造内の経路をノードとするネ

ットワーク構造の異同判定”, “Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値, EC 値）の利用”の一方または両方と組み合わせて使用される場合もあるので“グラフ同型判定のためのコアアルゴリズム”と呼ぶ．ここでは“グラフ同型判定のためのコアアルゴリズム”によるグラフ G_a とグラフ G_b の同型判定を説明する．

2.3.1 G_a と G_b

グラフ G_a を，$G_a=(V_a, E_a, L_a, \ell_a)$（$V_a$ は G_a の頂点の集合，E_a は G_a の辺の集合，L_a は G_a の頂点の属性の集合，ℓ_a は G_a の頂点に属性を対応付ける関数）とする．V_a に含まれる頂点に通し番号である頂点番号を付し，頂点番号が i の頂点を v_i と表す．頂点番号を行と列の番号として用いる G_a の隣接行列を A_a と表現する．A_a の i 行 j 列の成分の値は G_a に存在する辺(v_i, v_j)の数を示す．G_a の k 次隣接行列を A_a^k と定義する．A_a^k の i 行 j 列の成分の値は，G_a の頂点 v_i から G_a の頂点 v_j に至る k 個の辺を経由する経路の数を示す．G_a の頂点 v_i の属性は $\ell_a(v_i)$と表され，$L_a=\{\ \ell_a(v_i)\mid v_i\in V_a\ \}$である．$c\in L_a$ に対して $N_a(c)=\{\ v_i\in V_a\mid \ell_a(v_i)=c\ \}$とすれば，$F_a=\{\ (c, |N_a(c)|)\mid c\in L_a\ \}$が定まる．グラフ G_b に対して $V_b, E_b, L_b, \ell_b, A_b, A_b^k, N_b(c), F_b$ をグラフ G_a と同じように定める．

2.3.2 G_a と G_b の同型条件を満たす全単射

$|V_a|\neq|V_b|$または$|E_a|\neq|E_b|$の場合，G_a と G_b は同型ではないと判定される．$|V_a|=|V_b|$かつ$|E_a|=|E_b|$の場合は，F_a と F_b を比較する．$F_a\neq F_b$ の場合，G_a と G_b は同型ではないと判定される．$F_a=F_b$ の場合，“全単射 f: $V_a\rightarrow V_b$, $s.t.$ $(v_i, v_j)\in E_a\Leftrightarrow(f(v_i), f(v_j))\in E_b$, $\ell_a(v)=\ell_b(f(v))$が存在する”とき，G_a と G_b は同型であると判定される．以下では，“全単射 f: $V_a\rightarrow V_b$, $s.t.$ $(v_i, v_j)\in E_a\Leftrightarrow(f(v_i), f(v_j))\in E_b$, $\ell_a(v)=\ell_b(f(v))$”を満たす f を，“G_a と G_b の同型条件を満たす全単射”と表現する．$F_a=F_b$ の場合，G_a と G_b の同型条件を満たす全単射が存在するかどうかを調べる．

$F_a=F_b$ の場合，$L_a=L_b$ である．以下では，$L_a=L_b$ の要素すべてを含む配列を L とする．$L[i]\in L_a=L_b$ であり $N_a(L[i])$ を $V_a(i)$, $N_b(L[i])$を $V_b(i)$と表現する．コアアルゴリズムで後述の木構造の深さ優先探索を行う場合，配列 L の要素の配列順序は，G_a と G_b の同型条件を満たす全単射の探索の計算速度に影響を与え得る．配列 L の要素の配列順序の決め方については後述する．

写像 g に対し，$\{g(x)\mid x\in X\}$ を $g(X)$ と表すとき，G_a と G_b の同型条件を満たす全単射 f に対し，$V_b=f(V_a)$である．さらに $\ell_a(v)=\ell_b(f(v))$より $V_b(i)=f(V_a(i))$ $(i=1, 2,\cdots, |L|)$. したがって, 全単射 f_i: $V_a(i)\rightarrow V_b(i)$, $s.t.$ $f_i(v)=f(v)$が定義できる．G_a と G_b の同型条件を満たす全単射 f は $f_1, f_2,\cdots, f_{|L|}$に一意に分解される．これを$(f_1, f_2,\cdots, f_{|L|})=D(f)$と表現する．逆に $f=D^{-1}((f_1, f_2,\cdots, f_{|L|}))$であり，$(f_1, f_2,\cdots, f_{|L|})$から f が一意に定まる．$H_i'=\{\ h\mid h: V_a(i)\rightarrow V_b(i), h$ は全単射 $\}$とすると，$f_i\in H_i'$であり，$(f_1, f_2,\cdots, f_{|L|})\in H_1'\times H_2'\times\cdots\times H_{|L|}'$が成り立つので，$q\in H_1'\times H_2'\times\cdots\times H_{|L|}'$から定まる $p=D^{-1}(q)$に G_a と G_b の同型条件を満たす全単射があるかどうかを調べればよい．さらに，H_i'を全体集合としたときの集合$\{\ h\in H_i'\mid h\neq f_i$ が A_a と A_bから明らか $\}$の補集合 H_iに対して$f_i\in H_i$であり，$(f_1, f_2,\cdots, f_{|L|})\in H_1\times H_2\times\cdots\times H_{|L|}$が成り立つので，$q\in H_1\times H_2\times\cdots\times H_{|L|}$から定まる $p=D^{-1}(q)$に G_a と G_b の同型条件を満たす全単射があるかどうかを調べればよい．

H_i'から H_i を求めるための H_i が満たすべき条件については 2.3.10 で述べる．配列 L の要素の配列順序は，それにより定まる H_i の要素数$|H_i|$について$|H_1|\leqq|H_2|\leqq\cdots\leqq|H_{|L|}|$となるように定める．すなわち，暫定的に定めた配列順序の L から f_i, H_i', H_i を求めた後に，$|H_1|\leqq|H_2|\leqq\cdots\leqq|H_{|L|}|$となるよう，配列 L の並べ替えと f_i, H_i', H_i の番号付けの変更を行う．

2.3.3 G_a と G_b の同型条件を満たす全単射を探索するための木構造

G_a と G_b の同型条件を満たす全単射があるかどうかを調べるための木構造を生成する．木構造は，配列 L の要素の配列順序に依存して定まる．

root（深さ 0）に$|H_1|$個の子ノードを付加し，付加された$|H_1|$個の子ノード（深さ 1）に H_1 の異なる要素を 1 対 1 対応させる．次に，深さ 1 の各ノードにノードあたり$|H_2|$個の子ノードを付加し，付加された$|H_2|$個の子ノード（深さ 2）に H_2 の異なる要素を 1 対 1 対応させる．以下, 同じように，深さ i の各ノードにノードあたり$|H_{i+1}|$個の子ノードを付加し，付加された$|H_{i+1}|$個の子ノード（深さ i+1）に H_{i+1} の異なる要素を 1 対 1 対応させることを，i の値を 1 ずつ増加させながら，i+1≦$|L|$である間，繰り返す．

生成する木構造では，深さ i の各ノードに H_i の要素が対応する．木構造のノード x には，x の深さが i であれば H_i の要素が 1 つ対応する．このノード x に対応する H_i の要素を i の値に関わらず $h(x)$と表現する．木構造の深さ i のノードは，それが leaf でなければ，それぞれが，$|H_{i+1}|$個の子ノードを持つ．2 ノード x, y が共通の親ノードの子ノードである場合，$h(x)\neq h(y)$．したがって，木構造の深さ i のノードの総数は，$|H_1|\times|H_2|\times\cdots\times|H_i|$である．木構造の leaf の深さは，すべて，$|L|$に等しい．

x_j により木構造の深さ j のノードを表すとき，木構造の root から深さ i の特定のノード x_i に至る道筋は，root$\rightarrow x_1\rightarrow x_2\rightarrow\cdots\rightarrow x_i$ と表現される．この道筋を $r(x_i)$と呼ぶこととし，$r(x_i)$に $\alpha(x_i)=(h(x_1), h(x_2),\cdots, h(x_i))\in H_1\times H_2\times\cdots\times H_i$ および $\alpha(x_i)$により定まる $\beta(x_i)=D^{-1}(\alpha(x_i))=D^{-1}((h(x_1), h(x_2),\cdots, h(x_i)))$を対応させる．$h(x_i)$: $V_a(i)\rightarrow V_b(i)$, $h(x_i)$は全単射であるので，$\beta(x_i)$は $V_a(1)\cup V_a(2)\cup\cdots\cup V_a(i)$から $V_b(1)\cup V_b(2)\cup\cdots\cup V_b(i)$への全単射である．特に x_i が leaf（$i=|L|$）のとき $\beta(x_i)=\beta(x_{|L|})$は $V_a(1)\cup V_a(2)\cup\cdots\cup V_a(|L|)=V_a$ から $V_b(1)\cup V_b(2)\cup\cdots\cup V_b(|L|)=V_b$ への全単射である．また，$\alpha(x_{|L|})=(h(x_1), h(x_2),\cdots, h(x_{|L|}))\in H_1\times H_2\times\cdots\times H_{|L|}$であり，

逆に，任意の $q \in H_1 \times H_2 \times \cdots \times H_{|L|}$ に対して $q = \alpha(x_{|L|})$ となる $x_{|L|}$ が存在する．したがって，$q \in H_1 \times H_2 \times \cdots \times H_{|L|}$ から定まる $p = D^{-1}(q)$ に G_a と G_b の同型条件を満たす全単射があるかどうかを調べることは，すなわち，$\beta(x_{|L|})$ が G_a と G_b の同型条件を満たす全単射であるような leaf ノード $x_{|L|}$ があるかどうかを調べることである．

2.3.4 G_a と G_b の同型条件を満たす全単射であるための必要十分条件

要素数が $|V_a(i)| = |V_b(i)|$ に等しい配列 a，配列 b に対して，$V_a(i) = \{ v_m \in V_a \mid m \in a \}$，$V_b(i) = \{ v_m \in V_b \mid m \in b \}$ かつ（$\forall |V_a(i)| = |V_b(i)|$ 以下の自然数 j） G_a の頂点 $v_{a[j]}$ に対する $h(x_i)$ の像が G_b の頂点 $v_{b[j]}$ であるとき，配列 a，配列 b をそれぞれ $a(h(x_i))$，$b(h(x_i))$ と表す．

$V_a(1{:}i)$ により $V_a(1) \cup V_a(2) \cup \cdots \cup V_a(i)$ を，$V_b(1{:}i)$ により $V_b(1) \cup V_b(2) \cup \cdots \cup V_b(i)$ を表すものとする．要素数が $|V_a(1{:}i)| = |V_b(1{:}i)|$ に等しい配列 a，配列 b に対して，$V_a(1{:}i) = \{ v_m \in V_a \mid m \in a \}$，$V_b(1{:}i) = \{ v_m \in V_b \mid m \in b \}$ かつ（$\forall |V_a(1{:}i)| = |V_b(1{:}i)|$ 以下の自然数 j） G_a の頂点 $v_{a[j]}$ に対する $\beta(x_i)$ の像が G_b の頂点 $v_{b[j]}$ であるとき，配列 a，配列 b をそれぞれ $a(\beta(x_i))$，$b(\beta(x_i))$ と表す．

$M = \{ m_{ij} \}$ を $n \times n$ の行列，配列 u_r を $1, 2, \cdots, n$ から選んだ l_r 個の数の順列，配列 u_c を $1, 2, \cdots, n$ から選んだ l_c 個の数の順列とする（l_r, l_c は n 以下の自然数）．$l_r \times l_c$ の行列 $Y = \{ y_{ij} \}$ が $y_{ij} = m_{i'j'}$，$i' = u_r[i]$，$j' = u_c[j]$（i は l_r 以下，j は l_c 以下の自然数）を満たすようにつくられるとき Y を $M(u_r, u_c)$ と表す．配列 u が $(1, 2, \cdots, n)$ であるとき $M(u_r, u)$ を $M(u_r, :)$，$M(u, u_c)$ を $M(:, u_c)$ と表す．

これらの記法によると leaf ノード $x_{|L|}$ に対応する $\beta(x_{|L|})$ に対して $A_a(a(\beta(x_{|L|})), a(\beta(x_{|L|}))) = A_b(b(\beta(x_{|L|})), b(\beta(x_{|L|})))$ が成り立つことが，$\beta(x_{|L|})$ が G_a と G_b の同型条件を満たす全単射であるための必要十分条件である．

2.3.5 木構造の深さ優先探索による G_a と G_b の同型条件を満たす全単射算出の原理

$A_a(a(\beta(x_i)), a(\beta(x_i))) = A_b(b(\beta(x_i)), b(\beta(x_i)))$ のとき任意の自然数 k に対して $A_a^k(a(\beta(x_i)), a(\beta(x_i))) = A_b^k(b(\beta(x_i)), b(\beta(x_i)))$ が成り立つ．ノード x_p がノード x_c の親ノードであるとき $(\forall v \in a(\beta(x_p)))\ v \in a(\beta(x_c))$ と $(\forall v \in b(\beta(x_p)))\ v \in b(\beta(x_c))$ であるので，$A_a^k(a(\beta(x_c)), a(\beta(x_c))) = A_b^k(b(\beta(x_c)), b(\beta(x_c)))$ である場合，$A_a^k(a(\beta(x_p)), a(\beta(x_p))) = A_b^k(b(\beta(x_p)), b(\beta(x_p)))$ が成り立つ．したがって，leaf ノード $x_{|L|}$ に対して $A_a(a(\beta(x_{|L|})), a(\beta(x_{|L|}))) = A_b(b(\beta(x_{|L|})), b(\beta(x_{|L|})))$ が成り立てば $(\forall x_j \in \{ x \mid x \in r(x_{|L|})$ かつ $x \neq \text{root} \})$ $A_a^k(a(\beta(x_j)), a(\beta(x_j))) = A_b^k(b(\beta(x_j)), b(\beta(x_j)))$ が成り立つ．逆に，$(\exists k \in \{$自然数 $k \mid A_a^k(a(\beta(x_j)), a(\beta(x_j))) \neq A_b^k(b(\beta(x_j)), b(\beta(x_j)))\})$ ならば $(\forall x_{|L|} \in \{ x \mid x$ は x_j の子孫である leaf ノード$\})$ $A_a(a(\beta(x_{|L|})), a(\beta(x_{|L|}))) \neq A_b(b(\beta(x_{|L|})), b(\beta(x_{|L|})))$（$\beta(x_{|L|})$ が G_a と G_b の同型条件を満たす全単射ではない）が成り立つ．

このことから，$\exists k \in \{$number_of_Map 以下の自然数 $k \mid A_a^k(a(\beta(x_i)), a(\beta(x_i))) \neq A_b^k(b(\beta(x_i)), b(\beta(x_i)))\}$ ならば，ノード x_i の子孫を探索しない，木構造におけるノードの深さ優先探索により G_a と G_b の同型条件を満たす全単射が求まる．$\exists k \in \{$number_of_Map 以下の自然数 $k \mid A_a^k(a(\beta(x_i)), a(\beta(x_i))) \neq A_b^k(b(\beta(x_i)), b(\beta(x_i)))\}$ ならば，ノード x_i の子孫である leaf ノード $x_{|L|}$ に対して $A_a(a(\beta(x_{|L|})), a(\beta(x_{|L|}))) \neq A_b(b(\beta(x_{|L|})), b(\beta(x_{|L|})))$ が明らかだからである．

2.3.6 深さ優先探索のための木構造における root からノードに至る道筋とノードの配列表示

木構造における深さ優先探索のために木構造における root からノードに至る道筋とノードの配列表示を導入する．木構造の root から深さ i の特定のノード x_i に至る道筋 $r(x_i)$，root$\rightarrow x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_i$ における $x_1, x_2, \cdots, x_i$ にはそれぞれ $h(x_1), h(x_2), \cdots, h(x_i)$ が対応する．$h(x_1), h(x_2), \cdots, h(x_i)$ はそれぞれ $H_1, H_2, \cdots, Hi$ の要素である．Hj の要素に 1 から始まり $|Hj|$ で終わる要素番号を付すことにし，H_1 における $h(x_1)$ の要素番号を λ_1（$H_1[\lambda_1] = h(x_1)$），H_2 における $h(x_2)$ の要素番号を λ_2（$H_2[\lambda_2] = h(x_2)$），$\cdots$，Hj における $h(x_j)$ の要素番号を λ_j（$H_j[\lambda_j] = h(x_j)$），$\cdots$，Hi における $h(x_i)$ の要素番号を λ_i（$H_i[\lambda_i] = h(x_i)$）とすれば，配列 $(\lambda_1, \lambda_2, \cdots, \lambda_i)$ は $r(x_i)$ と 1 対 1 対応する．木構造のノードである x_i に対して $r(x_i)$ は 1 通りしかないので配列 $(\lambda_1, \lambda_2, \cdots, \lambda_i)$ は木構造のノード x_i とも 1 対 1 対応する．ノード x を $r(x)$ に対応する λ 値の配列で表すことにすると，$r(x_i)$ 上の root 以外のノード $x_1, x_2, \cdots, x_i$ はそれぞれ $r(x_i)$ に対応する配列 $(\lambda_1, \lambda_2, \cdots, \lambda_i)$ の部分配列である (λ_1)，(λ_1, λ_2)，$\cdots$，$(\lambda_1, \lambda_2, \cdots, \lambda_i)$ で表される．配列 Λ の 1 番目から j 番目までの部分の配列を $\Lambda[1{:}j]$ と表す記法を用い $\Lambda = (\lambda_1, \lambda_2, \cdots, \lambda_i)$ とすれば，x_1 は $\Lambda[1{:}1]$，x_2 は $\Lambda[1{:}2]$，$\cdots$，x_i は $\Lambda[1{:}i]$ と表現できる．以下の記述では，この記法を用いる，配列による木構造のノード x_i の表現および root から leaf ノード $x_{|L|}$ に至る道筋 $r(x_{|L|})$ の表現を行う．

2.3.7 木構造の深さ優先探索による G_a と G_b の同型条件を満たす全単射算出のための補助関数 next_comb

$B_i = \{ |H_i|$ 以下のすべての自然数 $\}$ とする．深さ優先探索を行うにあたり，$B_1 \times B_2 \times \cdots \times B_{|L|}$ の要素である配列 Cb の大小関係を定義する．2 つの異なる Cb である Cb_1 と Cb_2 に対し，$Cb_1[j] \neq Cb_2[j]$ となる j の最小値 j_0 を求める．このとき $Cb_1[j_0] < Cb_2[j_0]$ ならば Cb_1 は Cb_2 より小，$Cb_1[j_0] > Cb_2[j_0]$ ならば Cb_1 は Cb_2 より大と定義する．すると，すべての Cb を最小のもの（$|L|$ 個の 1 からなる配列）から始めて最大の配列 $(|H_1|, |H_2|, \cdots, |H_{|L|}|)$ まで順番に並べることができる．

深さ優先探索では，leaf ノード $x_{|L|}$ に至る道筋 $r(x_{|L|})$ に対応する配列 Cb を，探索が必要な道筋 $r(x_{|L|})$ に対応する Cb のみ，小さいものから順次，補助関数 next_comb により生成する．next_comb は上記の Cb の 1 つである Cb_j と $max\_comb = (|H_1|, |H_2|, \cdots, |H_{|L|}|)$ ならびに max_comb の要素

数 n＝|L|以下の自然数 d を与えて使う（ Cb＝next_comb(Cb_j, max_comb, d ））．d が n と等しいとき，Cb_j より大きい Cb が存在すればその中で最小のものを Cb として算出する．存在しなければ Cb を空配列として算出する．d が n と異なる（n よりも小さい）とき，Cb_j より大きい Cb であってその 1 番目から d 番目までの要素が Cb_j の 1 番目から d 番目までの要素と完全には一致しない Cb が存在すればその中で最小のものを Cb として算出する．存在しなければ Cb を空配列として算出する．

2.3.8 木構造の深さ優先探索による G_a と G_b の同型条件を満たす全単射算出の実際

計算の際に与える number_of_Map の値が 1 以上の整数値である場合に，以下の計算手順で，木構造の深さ優先探索による G_a と G_b の同型条件を満たす全単射算出を行う．この場合，number_of_Map は，A_a^k，A_b^k における k 値の上限値である．

Step 1. $issame$ = 0，Cb を|L|個の 1 からなる配列，max_comb = ($|H_1|$, $|H_2|$, …, $|H_{|L|}|$)とする．i = 1 とする．z_a と z_b を配列とし，z_a[1], z_b[1]に空配列を代入した後に Step 2 に進む．

Step 2. t = 1, d = i とする．ノード x を Cb[1:d]に対応するノードとする．a = z_a[d], b = z_b[d]として Step 3 に進む．

Step 3. a = [a, a(H_d[Cb[d]])], b = [b, b(H_d[Cb[d]])]とする．k = 1 とし，Step 4 に進む．a = a($\beta(x)$), b = b($\beta(x)$)である．ここで，配列 a の末尾に配列 a(H_d[Cb[d]])をつないできる配列を[a, a(H_d[Cb[d]])], 配列 b の末尾に配列 b(H_d[Cb[d]])をつないできる配列を[b, b(H_d[Cb[d]])]と表した．

Step 4. k＞number_of_Map であるときは Step 5 に進む．k ≦number_of_Map であるときは，A_a^k(a, a) = A_b^k(b, b)であれば k = k +1 として Step 4 を再度行う．A_a^k(a, a)≠A_b^k(b, b)であれば t = 0 として Step 5 に進む．

Step 5. t = 0 であるときは Step 6 に進む．t = 1 であるときは，z_a[d+1] = a, z_b[d+1] = b とする．d = |L|のときは Step 7 に進む（x は leaf である）．d＜|L|のときは，ノード x（Cb[1:d]に対応している）をその子ノードであって Cb[1:d+1] に対応するノードとする．その後に，d = d+1 として Step 3 に進む（このときノード x は Cb[1:d]に対応している）．

Step 6. Cb_0 = Cb, Cb = next_comb(Cb, max_comb, d)とする．Cb が空配列の場合は Step 8 に進む．Cb が空配列でない場合は，Cb[j]≠Cb_0[j]となる j の最小値を i として Step 2 に進む．

Step 7. $\beta(x)$が G_a と G_b の同型条件を満たす全単射であるので $issame$ = 1 とする．異同判定のみが目的であれば，$\beta(x)$を記録して Step 8 に進む．G_a と G_b の同型条件を満たす全単射すべての算出が目的であれば Step 6 に進む．

Step 8. $issame$ = 1 であれば，G_a と G_b は同型であると判定され，$issame$ = 0 であれば，G_a と G_b は同型ではないと判定される．

2.3.9 木構造の深さ優先探索によらない G_a と G_b の同型条件を満たす全単射算出

number_of_Map が 0 である場合に，以下の計算手順で，木構造の深さ優先探索によらない G_a と G_b の同型条件を満たす全単射算出を行う．

補助関数 next_comb により順次生成する Cb に対応する木構造の leaf ノード $x_{|L|}$に対して A_a(a($\beta(x_{|L|})$), a($\beta(x_{|L|})$))と A_b(b($\beta(x_{|L|})$), b($\beta(x_{|L|})$))を比較し，A_a(a($\beta(x_{|L|})$), a($\beta(x_{|L|})$)) = A_b(b($\beta(x_{|L|})$), b($\beta(x_{|L|})$))であるような leaf ノード $x_{|L|}$が見つかれば G_a と G_b は同型であると判定される（$issame$ = 1）．異同判定のみが目的であればここで計算を停止するが，G_a と G_b の同型条件を満たす全単射すべての算出が目的であれば，next_comb により生成する Cb に対応する leaf ノード $x_{|L|}$すべてについて A_a(a($\beta(x_{|L|})$), a($\beta(x_{|L|})$)) = A_b(b($\beta(x_{|L|})$), b($\beta(x_{|L|})$))となるものがないかを調べる．next_comb により生成する Cb に対応する leaf ノード $x_{|L|}$すべてについて A_a(a($\beta(x_{|L|})$), a($\beta(x_{|L|})$))≠A_b(b($\beta(x_{|L|})$), b($\beta(x_{|L|})$))であれば G_a と G_b は同型ではないと判定される（$issame$ = 0）．

2.3.10 H_i'から H_i を求めるための H_i が満たすべき条件

(h_1, h_2,…, $h_{|L|}$)＝D($\beta(x_{|L|})$)とするならば，A_a(a($\beta(x_{|L|})$), a($\beta(x_{|L|})$)) = A_b(b($\beta(x_{|L|})$), b($\beta(x_{|L|})$))が成り立つとき，(∀自然数 i≦|L|) A_a(a(h_i), a($\beta(x_{|L|})$)) = A_b(b(h_i), b($\beta(x_{|L|})$))かつ A_a(a($\beta(x_{|L|})$), a(h_i) = A_b(b($\beta(x_{|L|})$), b(h_i))であるので(∀自然数 i≦|L|) sum_c(A_a(a(h_i), :)) = sum_c(A_b(b(h_i), :))かつ sum_r(A_a(:, a(h_i))) = sum_r(A_b(:, b(h_i)))である．ここで，sum_c(M)は M のすべての列の和である列ベクトル，sum_r(M)は M のすべての行の和である行ベクトルである．また，A_a(a($\beta(x_{|L|})$), a($\beta(x_{|L|})$)) = A_b(b($\beta(x_{|L|})$), b($\beta(x_{|L|})$))が成り立つとき，(∀自然数 i≦|L|) A_a(a(h_i), a(h_i)) = A_b(b(h_i), b(h_i))である．したがって k，n を自然数とするとき，集合{ h_i∈H_i' | (∃k≦n) A_a^k(a(h_i), a(h_i))≠A_b^k(b(h_i), b(h_i))}，集合{ h_i∈H_i' | (∃k≦n) sum_c(A_a^k(a(h_i), :))≠sum_c(A_b^k(b(h_i), :)) または sum_r(A_a^k(:, a(h_i))) ≠ sum_r(A_b^k(:, b(h_i)))}，集合{ h_i∈H_i' | (∃k≦n) A_a^k(a(h_i), a(h_i))≠A_b^k(b(h_i), b(h_i))または sum_c(A_a^k(a(h_i), :)) ≠ sum_c(A_b^k(b(h_i), :)) または sum_r(A_a^k(:, a(h_i)))≠sum_r(A_b^k(:, b(h_i)))}は，h_i≠f_i が明らかな h_i∈H_i'の集合である．本稿のコアアルゴリズムでは n＝number_of_Map とし，集合{ h_i∈H_i' | (∃k≦n) A_a^k(a(h_i), a(h_i)) ≠A_b^k(b(h_i), b(h_i))}の補集合を H_i とした．すなわち H_i＝{ h_i∈H_i' | (∀k≦n) A_a^k(a(h_i), a(h_i))＝A_b^k(b(h_i), b(h_i))}である．

2.4 グラフのグラフ内経路を頂点とするグラフへの変換

概要および 1 に記した“ネットワーク構造内の経路をノードとするネットワーク構造の異同判定”における“ネットワーク構造内の経路をノードとするネットワーク構造”への化学量論的な代謝ネットワーク構造の変換の数学的表現である，グラフのグラフ内経路を頂点とするグラフへの変換について述べる．グラフ G を，G = (V, E, L, ℓ)（V は G の頂点の集合，E は G の辺の集合，L は G の頂点の属性の

集合，ℓ は G の頂点に属性を対応付ける関数）とする．V に含まれる頂点に通し番号である頂点番号を付し，頂点番号が i の頂点を v_i と表す．v_i に入る辺の数を $deg_{in}(v_i)$，v_i から出る辺の数を $deg_{out}(v_i)$，v_i と辺を形成する頂点の数を $deg(v_i)$と表す．$U = \{ v \in V \mid (deg_{in}(v) \neq 1$ または $deg_{out}(v) \neq 1)$ または $deg(v_i) = 1 \}$ が空集合でないグラフ G において，${}_pV = \{ p \mid p$ は $v_i \in U$ から $v_j \in U$ に至る $v \in U$ を経由しない経路を示す頂点配列 $(v_i, \cdots, v_j)\}$，${}_pE = \{ (p_i, p_j, v) \mid p_i \neq p_j$ である $p_i \in {}_pV$，$p_j \in {}_pV$ に対して $v \in p_i$ かつ $v \in p_j \}$，$p = (v_i, \cdots, v_j)$ のとき ${}_p\ell(p) = {}_p\ell((v_i, \cdots, v_j)) = (\ell(v_i), \cdots, \ell(v_j))$，${}_pL = \{ {}_p\ell(p) \mid p \in {}_pV \}$．$U$が空集合でないグラフ $G = (V, E, L, \ell)$から，グラフ ${}_pG$ が，${}_pG = ({}_pV, {}_pE, {}_pL, {}_p\ell)$（${}_pV$ は ${}_pG$ の頂点の集合，${}_pE$ は ${}_pG$ の辺の集合，${}_pL$ は ${}_pG$ の頂点の属性の集合，${}_p\ell$ は ${}_pG$ の頂点に属性を対応付ける関数）として得られる．

2.5 グラフの頂点の属性への拡張隣接関係値の追加

概要および 1 に記した“Morgan 法で算出される各ノードの拡張隣接関係値（extended connectivity 値, EC 値）の利用”のための，グラフの頂点の属性への拡張隣接関係値（extended connectivity 値, EC 値）の追加について述べる．

グラフ G を，$G = (V, E, L, \ell)$（V は G の頂点の集合，E は G の辺の集合，L は G の頂点の属性の集合，ℓ は G の頂点に属性を対応付ける関数）とする．グラフ G を，無向グラフとして，各頂点の EC 値を計算する［11］(58-61 頁)．得られた EC 値の集合を ${}_{EC}L$，G の頂点に EC 値を対応付ける関数を ${}_{EC}\ell$ とする．そして，$(\ell(v), {}_{EC}\ell\ (v)) \in L \times {}_{EC}L$ を $v \in V$ に対する EC 値追加属性とする．${}_{+EC}L = \{(\ell(v), {}_{EC}\ell\ (v)) \mid v \in V \}$ を EC 値追加属性の集合，$v \in V$ に対する ${}_{+EC}\ell$: $v \rightarrow (\ell(v), {}_{EC}\ell(v))$を G の頂点に EC 値追加属性を対応付ける関数とすると，グラフ G は，$(V, E, {}_{+EC}L, {}_{+EC}\ell)$とも表される．${}_{+EC}G = (V, E, {}_{+EC}L, {}_{+EC}\ell)$を G の EC 値追加属性表現と呼ぶ．

2.6 グラフ同型判定アルゴリズム *GCore*・${}_pGCore$・${}_{+EC}GCore$・${}_{+EC}{}_pGCore$

グラフ同型判定のためのコアアルゴリズム，グラフのグラフ内経路を頂点とするグラフへの変換，グラフの頂点の属性への拡張隣接関係値の追加を組み合わせることによりグラフ同型判定アルゴリズム *GCore*・${}_pGCore$・${}_{+EC}GCore$・${}_{+EC}{}_pGCore$ が生成する．それぞれ G_a と G_b のグラフ同型判定を下記のように行う．

2.6.1 *GCore*

G_a と G_b のグラフ同型判定をグラフ同型判定のためのコアアルゴリズムにより行う．

2.6.2 ${}_pGCore$

${}_pG_a$ と ${}_pG_b$ のグラフ同型判定を *GCore* により行い．G_a と G_b の同型条件を満たす全単射の候補を算出する．候補があり，かつ，それが G_a と G_b の同型条件を満たす全単射である場合に限り G_a と G_b が同型と判定する．

2.6.3 ${}_{+EC}GCore$

${}_{+EC}G_a$ と ${}_{+EC}G_b$ のグラフ同型判定を *GCore* により行う．

2.6.4 ${}_{+EC}{}_pGCore$

${}_pG_a$ と ${}_pG_b$ のグラフ同型判定を${}_{+EC}GCore$ により行い．G_a と G_b の同型条件を満たす全単射の候補を算出する．候補があり，かつ，それが G_a と G_b の同型条件を満たす全単射である場合に限り G_a と G_b が同型と判定する．

2.7 EFM 型経路の化学量論的な代謝ネットワーク構造算出アルゴリズムとグラフ同型判定アルゴリズムの一体化

EFM 型経路の化学量論的な代謝ネットワーク構造算出において，1 つの構造が算出されると，その構造を，それまでに算出されたすべての構造と，1 つ 1 つグラフ同型判定アルゴリズムにより異同判定しなければならない．グラフ同型判定アルゴリズムを実行する際には，*GCore*・${}_pGCore$・${}_{+EC}GCore$・${}_{+EC}{}_pGCore$ の場合は number_of_Map 次までの隣接行列の計算が，${}_pGCore$・${}_{+EC}{}_pGCore$ の場合は G の ${}_pG$ への変換が，${}_{+EC}GCore$・${}_{+EC}{}_pGCore$ の場合は EC 値の計算が必要であるが，EFM 型経路の化学量論的な代謝ネットワーク構造算出において，算出された構造に関するこれらの計算結果を構造とともに保持し，新たに算出される構造との異同判定に利用するよう，EFM 型経路の化学量論的な代謝ネットワーク構造算出アルゴリズムとグラフ同型判定アルゴリズムを一体化した．

3. 計算実験の結果と考察

糖質代謝モデルネットワーク［15,16］のペントースリン酸経路に含まれる反応の組み合わせにより 2 分子の fructose 6-phosphate と 1 分子の glyceraldehyde 3-phosphate から 3 分子の ribose 5-phosphate を生成する 5 反応（延べ 7 反応）からなる EFM 型経路ができる．これを EFM1 と呼ぶ． EFM1 に含まれる反応を，それぞれの数を n 倍（n は自然数）にして組み合わせると，$2 \times n$ 分子の fructose 6-phosphate と n 分子の glyceraldehyde 3-phosphate から $3 \times n$ 分子の ribose 5-phosphate を生成する $5 \times n$ 反応（延べ $7 \times n$ 反応）からなる EFM 型経路ができる．これを EFMn と呼ぶ．また，EFM 型経路の unique なネットワーク構造を EFM 型経路の“variant”と呼ぶ．本項の計算実験では，多数回の異同判定を要する EFM2, EFM3, EFM4, EFM5 の variant の算出を Matlab 上で行った（Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99GHz）．

表 1 に，EFM2 の variant をグラフ同型判定アルゴリズム *GCore*・${}_pGCore$・${}_{+EC}GCore$・${}_{+EC}{}_pGCore$ と一体化した EFM 型経路の化学量論的な代謝ネットワーク構造算出アルゴリズムを用いて計算した結果を示す．*GCore* と${}_{+EC}GCore$ では，number_of_Map 値の 0 から 1 への増加により計算時間が大きく短縮された．*GCore* では，number_of_Map 値の 1 から 2 への増加により計算時間が大きく短縮された．number_of_Map 値が 1 以上 3 以下のときの計算時間は ${}_{+EC}GCore$，${}_pGCore$，${}_{+EC}{}_pGCore$ すべて 1 秒前後であったが，number_of_Map 値 2 と 3 の比較，${}_pGCore$ と${}_{+EC}{}_pGCore$ の比

較から，number_of_Map の増加と ${}_pG$ から ${}_{+EC}{}_pG$ の計算には相応の計算コストが必要であることが示唆された．

表 1　EFM2 の variant の算出

number_of_Map	計算時間（秒）			
	GCore	${}_{+EC}$*GCore*	${}_p$*GCore*	${}_{+EC}{}_p$*GCore*
0	>120	>120	1.79	1.32
1	46.09	1.13	1.22	1.31
2	14.12	0.97	1.20	1.27
3	15.54	0.99	1.22	1.32

表 2　EFM2, EFM3, EFM4, EFM5 の variant の算出 1

経路	number_of_Map	計算時間（秒）		
		${}_{+EC}$*GCore*	${}_p$*GCore*	${}_{+EC}{}_p$*GCore*
EFM2	3	0.99	1.22	1.32
EFM3	3	15.48	19.29	19.64
EFM4	3	250.32	273.77	239.69
EFM5	3	4603.55	5550.13	2543.77

表 3　EFM2, EFM3, EFM4, EFM5 の variant の算出 2

経路	ノードの数*	variant の数	異同判定の回数
EFM2	42	8	612
EFM3	63	26	8146
EFM4	82	94	114342
EFM5	102	326	1714915

* 仮想原料反応と仮想目的反応のノードを含む反応ノードの個数と代謝産物ノードの個数の和である．

表 2 に，EFM2, EFM3, EFM4, EFM5 の variant 算出の計算時間（number_of_Map 値 3）を示す．最も計算時間が短いのは，EFM2 と EFM3 では ${}_{+EC}$*GCore*，反応数が多い EFM4 と EFM5 では ${}_{+EC}{}_p$*GCore* であった．

表 3 に，EFM2, EFM3, EFM4, EFM5 の，1 つの variant を構成するノードの数，算出された unique な variant の数，unique な variant 算出に必要であった異同判定の回数を示す．

本稿では，EFM 型代謝経路の化学量論的な代謝ネットワーク構造の算出のための異同判定アルゴリズムの最適化を試みた．調べたアルゴリズムのうち ${}_{+EC}$*GCore*，${}_p$*GCore*，${}_{+EC}{}_p$*GCore* は，1 以上の number_of_Map で用いたとき，それぞれ，計算時間の短縮に有効であった．計算時間の短縮に最も有効なものは，EFM 型経路により異なっていた．しかしながら，得られた結果は，含まれる反応の数の多い EFM 型経路の化学量論的な代謝ネットワーク構造の算出においては，*G* から ${}_pG$ を，${}_pG$ から ${}_{+EC}{}_pG$ を計算することには相応の計算コストが必要であるものの，本稿で調べたアルゴリズムのうち ${}_{+EC}{}_p$*GCore* の使用を勧めているように思われる．

参考文献

[1] Schuster, S.; Hilgetag, C. On elementary flux modes in biochemical reaction systems at steady state. *J. Biol. Syst.* **1994**, *2*, 165-182.
[2] Schilling, C. H.; Palsson, B.O. The underlying pathway structure of biochemical reaction networks. *Proc. Nat. Acad. Sci. USA* **1998**, *95*, 4193-4198.
[3] 太田 潤　代謝ネットワークにおける elementary flux mode 型経路の完全原子レベルマッピング．情報処理学会研究報告, **2019**, Vol. 2019-BIO-59, No. 2, 1-6.
[4] 太田 潤　Elementary flux mode 型代謝経路の完全原子レベルマッピングの高速化．情報処理学会研究報告, **2019**, Vol.2019-BIO-60, No. 6, 1-7.
[5] 太田 潤　対称性のある代謝産物を通る elementary flux mode 型代謝経路の完全原子レベルマッピング．情報処理学会研究報告, **2020**, Vol.2020-BIO-61, No. 9, 1-8.
[6] Acuña, V.; Milreu, P.V.; Cottret, L.; Marchetti-Spaccamela, A.; Stougie, L.; Sagot, M.F. Algorithms and complexity of enumerating minimal precursor sets in genome-wide metabolic networks. *Bioinformatics* **2012**, *28*, 2474-83.
[7] Ravikrishnan, A.; Nasre, M.; Raman, K. Enumerating all possible biosynthetic pathways in metabolic networks. *Sci. Rep*. **2018**, *8*, 9932. DOI: 10.1038/s41598-018-28007-7
[8] 太田 潤　Elementary flux mode 型代謝経路の化学量論的に釣り合った代謝産物レベルネットワーク構造の算出．情報処理学会研究報告, **2020**, Vol.2020-BIO-62, No. 3, 1-6.
[9] 太田 潤　Elementary flux mode 型代謝経路の化学量論的に釣り合った代謝産物レベルネットワーク構造算出の高速化．情報処理学会研究報告, **2020**, Vol.2020-BIO-63, No. 1, 1-7.
[10] 太田 潤　Elementary flux mode 型代謝経路の化学量論な代謝ネットワーク構造の算出: 算出されたネットワーク構造間の異同判定の高速化．情報処理学会研究報告, **2020**, Vol.2020-BIO-64, No. 5, 1-7.
[11] Gasteiger, J.; Engel, T.; 船津公人; 佐藤寛子; 増井秀行　ケモインフォマティクス - 予測と設計のための化学情報学．丸善株式会社，**2005**.
[12] Morgan, H. L. The generation of a unique machine description for chemical structures – A technique developed at Chemical Abstracts Service. *J. Chem. Doc.* **1965**, *5*, 107-113.
[13] Ullmann, J. R. An algorithm for subgraph isomorphism. *J. Assoc. Comput. Mach.* **1976**, *23*, 31-42.
[14] Carletti, V.; Foggia, P.; Saggese, A.; Vento, M. Challenging the time complexity of exact subgraph isomorphism for huge and dense graphs with VF3. *IEEE trans. pattern anal. mach. Intel.* **2018**, *40*, 804-818.
[15] Ohta, J. Connectivity matrix method for analyses of biological networks and its application to atom-level analysis of a model network of carbohydrate metabolism. *IEE Proc. Syst. Biol.* **2006**, *153*, 372-374. DOI: 10.1049/ip-syb:20060018
[16] Ohta, J. Single-atom tracing in a model network of carbohydrate metabolism and pathway selection. *IPSJ Transactions on Bioinformatics* **2018**, *11*, 1-13.