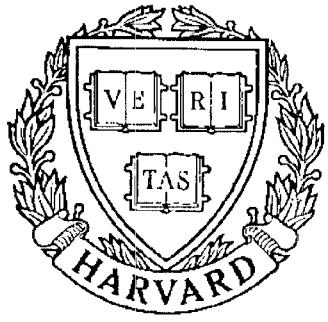
TECHNICAL RESEARCH REPORT

SYSTEMS RESEARCH CENTER



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

Computed-Aided Synthesis of Biochemical Pathways

by M.L. Mavrovouniotis

TR 90-42

COMPUTER-AIDED SYNTHESIS OF BIOCHEMICAL PATHWAYS

Michael L. Mavrovouniotis
Systems Research Center and Chemical Engineering
University of Maryland
College Park, MD 20742

ABSTRACT

The synthesis of biochemical pathways satisfying stoichiometric constraints is discussed. Stoichiometric constraints arise primarily from designating compounds as required or allowed reactants, and required or allowed products of the pathways; they also arise from similar restrictions on intermediate metabolites and bioreactions participating in the pathways. An algorithm for the complete and correct solution of the problem is presented; the algorithm satisfies each constraint by recursively transforming a base-set of pathways. The algorithm is applied to the problem of lysine synthesis from glucose and ammonia. In addition to the established synthesis routes, the algorithm constructs several alternative pathways that bypass key enzymes, such as *malate dehydrogenase* and *pyruvate dehydrogenase*. Apart from the construction of pathways with desired characteristics, the systematic synthesis of pathways can also uncover fundamental constraints in a particular problem, by demonstrating that no pathways exist to meet certain sets of specifications. In the case of lysine, the algorithm shows that oxaloacetate is a necessary intermediate in all pathways leading to lysine from glucose, and that the yield of lysine over glucose cannot exceed 67% in the absence of enzymatic recovery of carbon dioxide.

Keywords: metabolic pathway, biochemical pathway, pathway, design, synthesis

INTRODUCTION AND BACKGROUND

The synthesis of biochemical pathways involves the construction of pathways, i.e., sets of enzyme-catalyzed bioreactions whose stoichiometry is given, to meet certain specifications[1]. A class of specifications can be formulated by classifying each available building block, i.e., each metabolite and each bioreaction, according to the role it can play in the synthesized pathways. For example, a set of specifications may include some metabolites designated as required final products of the pathways, other metabolites as allowed reactants or by-products, and some bioreactions as prohibited from participating in the pathways.

Systematic synthesis of pathways that satisfy a set of such specifications is relevant in the early steps of the conception and design of a bioprocess, where a pathway must be chosen for the production of the desired product. With a method for the construction of all biochemical pathways satisfying a set of constraints, all possible alternative pathways can be constructed and a better informed decision can be made. Furthermore, the existence of common characteristics shared by many pathways, such as a fixed intermediate metabolite or dependence of the yield on a certain bioreaction, allows the identification of fundamental limitations in the process. For the synthesis of desired bioproducts, these limitations are crucial factors in the feasibility of the process and the selection of appropriate strains.

Such fundamental limitations are also important in the *catabolism*, for the identification of a mutant strain lacking a particular enzyme. One needs to identify those sets of substrates on which the mutant microbe (lacking a particular bioreaction) should grow, and those sets of substrates on which the microbe should *not* grow. The ability of the microbe to grow depends primarily on the presence of suitable pathways to consume the substrates in question. Especially for mutant strains, such pathways may differ significantly from the standard and well known routes. Thus, systematic and

complete generation of pathways is a much more reliable way to predict the ability of a mutant strain to grow on specified sets of substrates. Consequently, it can have a significant impact on the identification of mutant strains lacking a certain bioreaction. Conversely, if the target is not elimination of an enzyme but absence of growth on specific sets of substrates, one must pinpoint the enzymes that should be eliminated to *block* all the pathways for the catabolism of the substrates. The selection of an appropriate set of enzymes depends on the correct generation of all relevant pathways.

The first effort for systematic synthesis of biochemical pathways was made by Seressiotis and Bailey[2]. They presented an approach for synthesizing pathways that start from a given substrate and produce a target product. In their approach, one starts with all the different reactions that can consume the designated substrate. Each of these reactions produces certain products, which become available for use by other reactions. Pathways are thus constructed recursively, proceeding from the substrate towards the product. For any pathway being expanded, the available substrates include all products that the pathway has produced up to that point. The addition of reactions that lead from these substrates to new products creates new pathways that produce additional products. Seressiotis and Bailey[2] improved this basic procedure by addressing some of its complications. They introduced means for the elimination of certain redundant pathways and, through special handling of currency metabolites, they achieved a partial reduction of the high computational complexity inherent in the approach. The method, however, still has a number of fundamental shortcomings:

- The computer program which synthesizes pathways is not accompanied by a well-defined formal algorithm.
- The formulation of the problem is restrictive. It involves the specification of only one required reactant and one required product and no further refinement in terms of *allowed* reactants and by-products.
- While all pathways constructed by the program are correct (i.e., they involve the required substrate and the required product), there exist pathways which the method cannot synthesize[2]. The exact set of pathways that the program can synthesize is not explicitly described.
- The computational complexity of the approach prohibits its application to large bioreaction networks which involve products far removed from their substrates. This disadvantage is compounded by the fact that the method does not yield significant partial results: When it cannot run to completion, it does not offer any insights on how the particular synthesis problem could be better formulated.

A different approach for the synthesis of biochemical pathways is presented here. It is based on a novel algorithm which, through a radically different formalization and solution of the problem[1], overcomes the restrictions of the previous efforts. The next section gives an overview of the precise formulation of the problem and the developed algorithm, which is complete, correct, and computationally tractable. A case study on the biosynthesis of lysine from glucose and ammonia is then examined. Several non-obvious alternative pathways are constructed, such as pathways bypassing *malate dehydrogenase*. Based on the results of the algorithm, two important constraints on the process are identified, stating that all pathways must involve oxaloacetate as an intermediate, and that the yield of lysine over glucose depends on bioreactions recovering carbon dioxide. The mathematical properties of the algorithm, along with the results of the case study, show that there is great value in the proposed approach to systematic pathway synthesis. The work presented here is part of a broader effort for methodic synthesis and analysis of biochemical pathways[1].

OVERVIEW OF THE ALGORITHM

Constraints on metabolites. A given metabolite can participate in a pathway in any of three capacities: (a) as a net *reactant* or substrate of the pathway; (b) as a net *product* of the pathway; and (c) as an *intermediate* in the pathway, i.e., participating without *net* consumption or production. One can impose constraints on pathways by stating which metabolites are *required* and which are *prohibited* to participate in the synthesized pathways in each of the above three capacities. Not all

metabolites need be strictly constrained as required or prohibited. Some may simply be *allowed* to participate in the pathways.

For example, metabolites (from a database of biochemical reactions and metabolic intermediates) can be classified according to whether they are allowed to be net **reactants** in the pathways: **(1)** *Required reactants* (or desired reactants) *must* be consumed by the pathway; **(2)** *allowed reactants* may or may not be consumed by the pathway; and **(3)** *excluded reactants* (or prohibited reactants) *must not* be consumed by the pathway. In a realistic synthesis problem, the default characterization for each metabolite is specification (3): The bulk of the metabolites in the database are *excluded* from being net reactants of the synthesized pathways.

Specification (1) underlies a strict inequality, i.e., stoichiometric coefficient of the metabolite (in the pathway) less than zero, while specification (2) underlies a loose inequality, i.e., stoichiometric coefficient less than or equal to zero. Thus, the first constraint is strict, while the second one is loose. This distinction is relevant in the description of the algorithm, because strict constraints are initially satisfied only in their loose form.

The classification of metabolites as potential products or intermediates[§] of the pathways is quite similar. For intermediates, however, the default characterization differs, as most of the metabolites would normally be classified as *allowed* intermediates. The constraints on different roles of the same metabolite are not independent. For example, a metabolite that is required as a net product *must* be excluded as a reactant.

Constraints on bioreactions. A given bioreaction can participate in pathways in either (a) its *forward,* or (b) its *backward* direction. Thus, one can impose constraints by stating which bioreactions are required, which are allowed, and which are prohibited to participate in the synthesized pathways in each of the two directions. Many constraints designating bioreactions excluded in the backward direction will be present, stemming from knowledge about the (thermodynamic or mechanistic) irreversibility of bioreactions.

The constraints on bioreactions are not independent. For example, a bioreaction required in the forward direction *must* excluded in the backward direction.

Nature of the algorithm. The algorithm that will be presented in this paper is devoted to the satisfaction of above kinds of constraints imposed on the participation of metabolites and bioreactions in biochemical pathways. These constraints are in essence *stoichiometric*, and they could be formulated to refer, quantitatively, to the stoichiometries of reactions and pathways.

Given a set of stoichiometric constraints and a database of biochemical reactions, the developed algorithm synthesizes all biochemical pathways satisfying the stoichiometric constraints. The algorithm is based on the *iterative* satisfaction of constraints, and the stepwise transformation of the initial set of available bioreactions (which can be thought of as one-step pathways that, in general, do not satisfy the constraints), into a final set of pathways, which satisfy all imposed constraints. The algorithm consists of three phases, described below. More details on the operation of the algorithm are given in Appendix A. Appendix B provides an example of a step-by-step application of the algorithm. Appendix C briefly discusses theoretical and practical issues regarding the computational complexity of the problem.

Reaction-processing phase. In order to account for the reversibility of reactions, the inverses of the original reactions are created as independent reactions. From this point on, we prohibit the participation of both the forward and the reverse reaction in the same pathway, because such a pathway would be redundant.

The constraints placed on the backward direction of each of the original reactions are then easily transformed into constraints on the new (reverse) reactions. Constraints referring to required reactions are strict, and they are not processed at this preliminary stage. Their satisfaction is achieved in the last phase of the algorithm. However, constraints dictating that certain reactions are excluded

---

§ Constraints on intermediates are generally not motivated by physiological considerations. They are usually a device for selecting a particular subset of the synthesized pathways.

from the constructed pathways can be satisfied right from the start. Such reactions are simply eliminated and removed from the active database.

The remaining reactions can be thought of as *one-step pathways* which in general do not satisfy the constraints imposed on metabolites, but do satisfy the loose constraints imposed on bioreactions. These initial pathways will be combined in subsequent phases of the algorithm to form longer and longer pathways satisfying more and more constraints.

Metabolite-processing phase. The main body of the algorithm tackles one constraint at a time, by transforming the set of pathways (which at the beginning is the same as the set of available bioreactions). Thus, at each iteration stage in the synthesis algorithm, the problem state (or the *state of the design*[3,4]) is characterized by a set of partial pathways (satisfying the constraints that have already been processed) and a set of stoichiometric constraints which have yet to be satisfied. Each constraint corresponds to a particular metabolite that still remains to be processed.

At each pathway-expansion step, one of the remaining constraints is chosen as a goal. The most suitable metabolite is the one which participates (as a reactant or product) in the smallest possible number of pathways that are active in the current state of the problem. The set of active pathways is then modified to satisfy the constraint. For example, if the constraint designates a metabolite as an excluded reactant and excluded product, all possible combination-pathways must be constructed by combining one pathway consuming the metabolite and one pathway producing it, such that the metabolite is eliminated from the overall net stoichiometry. As was noted earlier, we cannot combine two pathways if they involve the same reaction in different directions. Once the combinations are constructed, all pathways consuming or producing the metabolite are deleted, because they violate the constraint.

If, on the other hand, the metabolite is an excluded reactant but a required product, the same combination-pathways are constructed, as before, but only the pathways consuming the metabolite are deleted. The pathways producing the metabolite satisfy the constraint in its strict form, and they are retained. By their construction, the newly-created combination pathways satisfy the constraint only in its loose form (i.e., the constraint stating the metabolite is an *allowed*, rather than *required*, product), but this is acceptable at this phase of the algorithm. The satisfaction of the constraint in its strict form will receive additional attention in the pathway-marking phase.

As illustrated by the two examples above, two subsets of the current pathway set must generally be assembled: The list of pathways that produce the metabolite and the list of pathways that consume the metabolite. Pathways may, at this step of the algorithm, be **constructed** as linear combinations of precisely one pathway from the first list and precisely one pathway from the second list. Pathways from the two lists may be **deleted** if they do not satisfy the loose form of the constraint at hand. More details on the operation of the algorithm are given in Appendix A. Appendix B clarifies the application of the algorithm using an easy example.

The linear nature of the constraints has an important consequence. Once a constraint is satisfied by all surviving pathways, further linear combinations of the surviving pathways (constructed in later stages) will never violate the constraint. Thus, after the processing of each constraint, the new active pathways satisfy *all previously processed constraints* — at least in their loose form.

Pathway-marking phase. At the end of the metabolite-processing phase, there is a set of pathways satisfying, in their loose form, all the constraints imposed at the beginning. Because of the linear nature of the constraints, all linear combinations of pathways also satisfy the constraints, in the same loose form.

Some pathways from the final set also satisfy a subset of the original strict constraints. Combinations of pathways (with non-negative coefficients) satisfy the *union* of such constrains satisfied by their constituent pathways. Thus, by marking each pathway with the strict-inequality constraints it satisfies, the final answer to the synthesis problem is obtained: The pathways satisfying the original stoichiometric constraints are all those linear combinations of pathways from the final set which have at least one constituent pathway satisfying each of the strict constraints. Appendix B provides an example of a step-by-step application of the algorithm.

Properties of the algorithm. The algorithm possesses provable mathematical properties[1]. The algorithm is correct because it generates *only* feasible pathways, i.e., pathways satisfying the stoichiometric constraints imposed. The algorithm is complete because it does not miss any solutions, i.e., it generates (a description of) *all* pathways satisfying the constraints. For pathways of fixed maximum size, the computational complexity of the algorithm is polynomial with respect to the size of the reaction database.

The algorithm was implemented in Symbolics Common LISP. The performance of the implementation of the algorithm greatly varies with the exact formulation of the problem, because the number of solutions (final set of pathways) depends on subtle points of the problem formulation[1]. The rough requirements for a typical problem with 220 reactions and 400 metabolites generating 5000 pathways, are 8 minutes of elapsed time and 1.7M of allocated words (roughly 8 Mbytes) on a Symbolics 3650 computer. Our experience has shown that the algorithm is reasonably efficient; it will run in a short time (of the order of a few minutes) for carefully formulated problems. Appendix C briefly discusses theoretical and practical issues regarding the computational complexity of the problem.

A CASE STUDY

The main focus of this paper is a demonstration of the value of the algorithm through a case study on the synthesis of lysine from glucose and ammonia. It should be emphasized that the analysis performed in the case study is not exhaustive. The aim of the analysis is not to provide a complete enumeration of the pathways for the biosynthesis of lysine, but merely to demonstrate the utility of the synthesis methodology. Furthermore, lysine possesses no particular qualities (positive or negative) in relation to the synthesis methodology. Thus, the performance of the method in examining the synthesis of other bioproducts is not expected to vary radically from its performance in the case of lysine.

The basic procedure followed in this case study involves the initial construction of a generally accepted pathway, and subsequent exploration of alternatives that omit key enzymes or intermediates. This exploration leads to the identification of fundamental constraints on the structure and yield of lysine-producing pathways.

It is not possible to present all pathways synthesized by the algorithm in this paper because several hundred pathways are constructed (the exact number depending on details of the database and the formulation of the problem). Note the set of pathways constructed is only complete with respect to the (necessarily incomplete) database of bioreactions that is used; introduction of additional bioreactions in the database makes the results of the previous search incomplete. The particular pathways that are synthesized, however, remain valid, and can be useful regardless of the elusive completeness of the search. This section focuses on various alternative routes that bypass specific enzymes or intermediates. Although it is significant that the developed algorithm does indeed construct all possible pathways (in contrast to another known approach[2] which is incomplete). This does not mean, however, that one has to examine all pathways at once; each alternative route presents a distinct possibility that is relevant in examining fluxes in the metabolism or routes for the synthesis of a bioproduct.

A Basic Pathway for the Production of Lysine. The core of the initial bioreaction network shown in Figure 1 includes several basic pathways, including glycolysis, the citric acid cycle (which will be referred to as TCA), the bacterial pathway from oxaloacetate to aspartate and on to lysine, and the enzymes *Lactate dehydrogenase*, *Glutamate dehydrogenase*, and *Glutamine synthetase*. The figure has been drawn in a simplified form, as many side-reactants and side-products are not shown, and many reactions are lumped together (for example, the arrow drawn from aspartate-semialdehyde to lysine represents 6 individual bioreactions).

In order to simplify the case study, the enzyme *α-ketoglutarate dehydrogenase* is assumed non-functional and has not been included; the glyoxylate shunt complements TCA and makes up for the

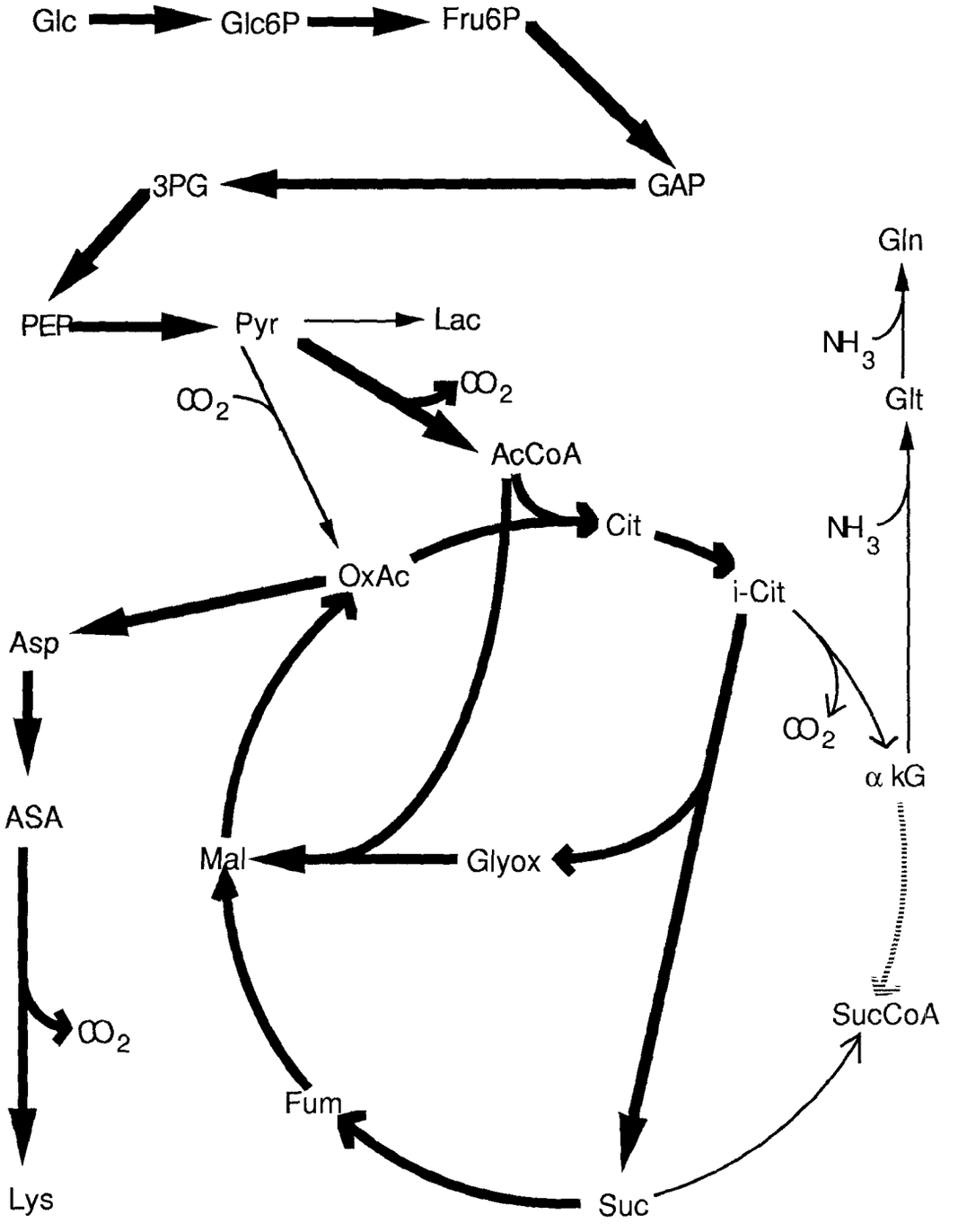


Figure 1: The basic bioreaction network for the synthesis of lysine. With bold arrows, a partial pathway which is the normal route for the production of lysine from glucose through the glyoxylate shunt.

absence of *α-ketoglutarate dehydrogenase*. This network serves as a frame of reference, because it contains all the bioreactions one would initially take into account. One of the purposes of the case

study is to demonstrate that many bioreactions that have *not* been included in the network could play a significant role in lysine-producing pathways.

Within the reference bioreaction network, one of the basic pathways believed to function in bacteria (such as *Brevibacterium Flavum*) for the conversion of glucose to lysine is shown by the thick reaction-arrows in Figure 1. This pathway is actually only a partial one, because the segment leading from aspartate to lysine requires succinyl-CoA and glutamate (at the same time producing succinate and a-ketoglutarate). The pathway can be completed by incorporating additional reactions to balance the stoichiometries for succinyl-CoA, glutamate, succinate, and a-ketoglutarate. The completed pathway, shown in Figure 2, balances succinyl-CoA, glutamate, succinate, and a-ketoglutarate. The reactions added are *succinate kinase* and *glutamate dehydrogenase*, which comprise the simplest alternative. While the solution is simple in this particular case, more complicated sets of reactions might need to be added in other cases (where more unusual side-reactants and side-products may be involved). The algorithm would also be more important if other, more complicated ways for completing the pathway were of interest.

The stoichiometries of currency metabolites (such as ATP or NAD) remain unbalanced, but such currency metabolites will be considered allowed reactants and allowed products. This is necessary because their stoichiometries can be balanced through a very large number of reactions, and an effort in that direction would be hindered by combinatorial explosion.

If a particular bioreaction or segment is identified as a bottleneck in the basic pathway, the synthesis can be focused on pathways that bypass the bottleneck. As a first example, the algorithm will be used to generate alternative pathways that bypass *malate dehydrogenase*, which is hypothesized to be the pathway's bottleneck.

Figure 3 shows a first pathway that excludes *malate dehydrogenase*. This pathway in fact bypasses the whole TCA cycle through the direct carboxylation of pyruvate into oxaloacetate. This can be achieved by either *Pyruvate carboxylase* or *Oxaloacetate decarboxylase*. The pathway also has a more attractive maximum molar yield. If we neglect restrictions stemming from reduction balances and focus only on the carbon-skeleton, the maximum yield of the pathway is 100%, i.e., the pathway yields one mole of lysine per mole of glucose, as compared to a molar yield of 67% for the initial pathway of Figure 2.

If the original pathway has some good traits, one might prefer to bypass only the immediate vicinity of the bottleneck and retain much of the structure of the original pathway intact, including the TCA cycle. A first alternative, shown in Figure 4, involves bypassing *malate dehydrogenase* with a set of just two reactions: *Lactate-Malate transhydrogenase* achieves the conversion

$$\text{malate} + \text{pyruvate} \rightarrow \text{oxaloacetate} + \text{lactate}$$

while *Lactate dehydrogenase* achieves the conversion:

$$\text{lactate} \rightarrow \text{pyruvate}$$

The combination of the two reactions converts malate to oxaloacetate. It is interesting to note that this pathway uses *lactate dehydrogenase* in the direction opposite to that originally drawn in Figure 1.

Another alternative, shown in Figure 5, involves:

- Conversion of malate to fumarate by using *Fumarase* in the direction opposite to that initially assumed in Figure 1
- Conversion of succinate to fumarate by *Succinate dehydrogenase* as in the original pathway
- Conversion of fumarate into aspartate through *Aspartate aminolyase*

Since oxaloacetate is used in order to form citrate, half of the aspartate must be recycled back into oxaloacetate to close the TCA loop. In the pathway of Figure 5 the reaction *aspartate glutamate transaminase* converts aspartate to oxaloacetate, by operating in the direction reverse to that assumed in the original bioreaction network (Figures 1 and 2).

A small variation in this pathway is created if, for the conversion of oxaloacetate to aspartate, a set of two reactions is used (Figure 6). First, *Glycine-oxaloacetate aminotransferase* converts glyoxylate and aspartate into glycine and oxaloacetate. Second, *Glycine dehydrogenase* recycles glycine into glyoxylate.

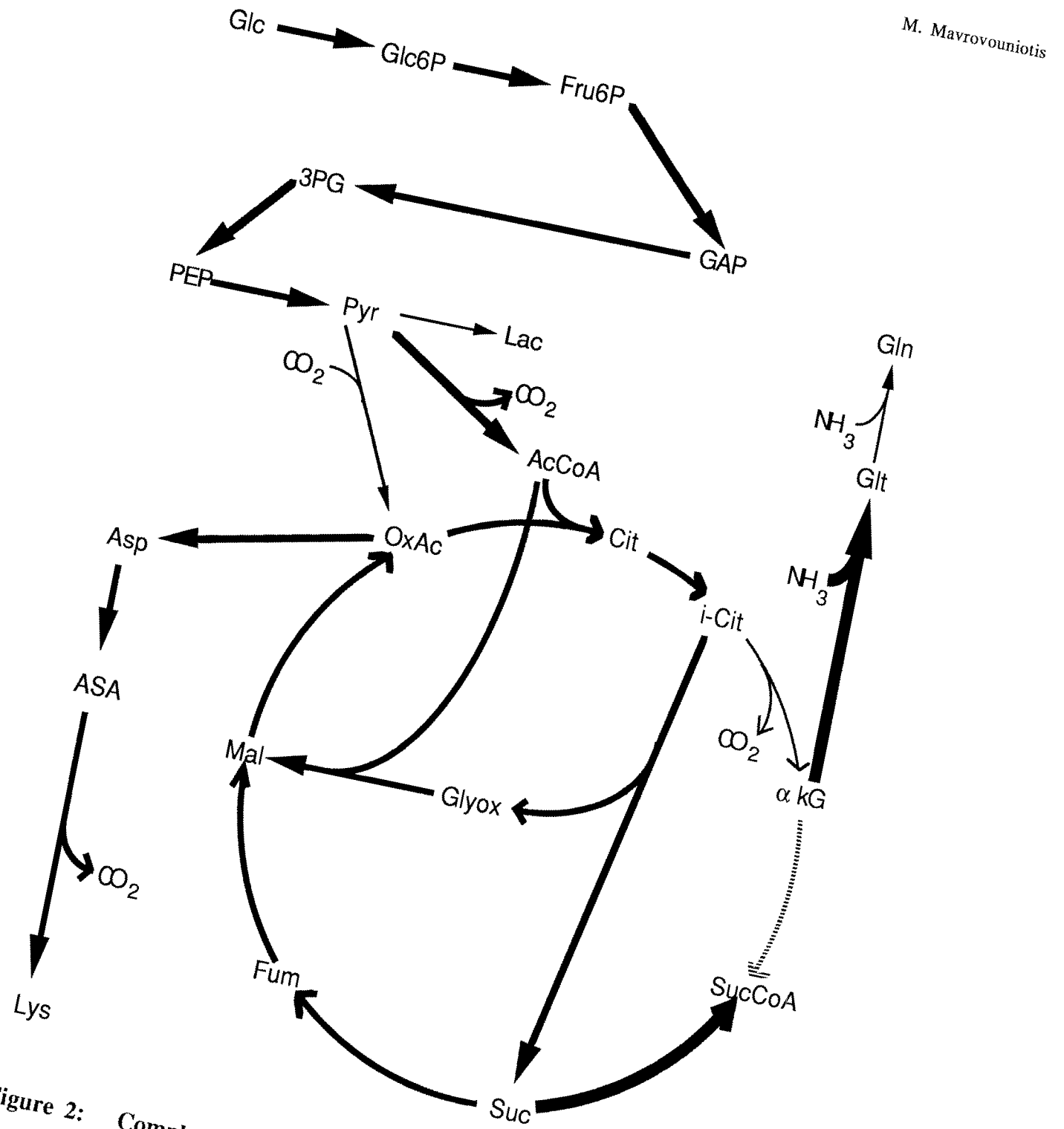


Figure 2: **Completion of the basic pathway for the synthesis of lysine.**

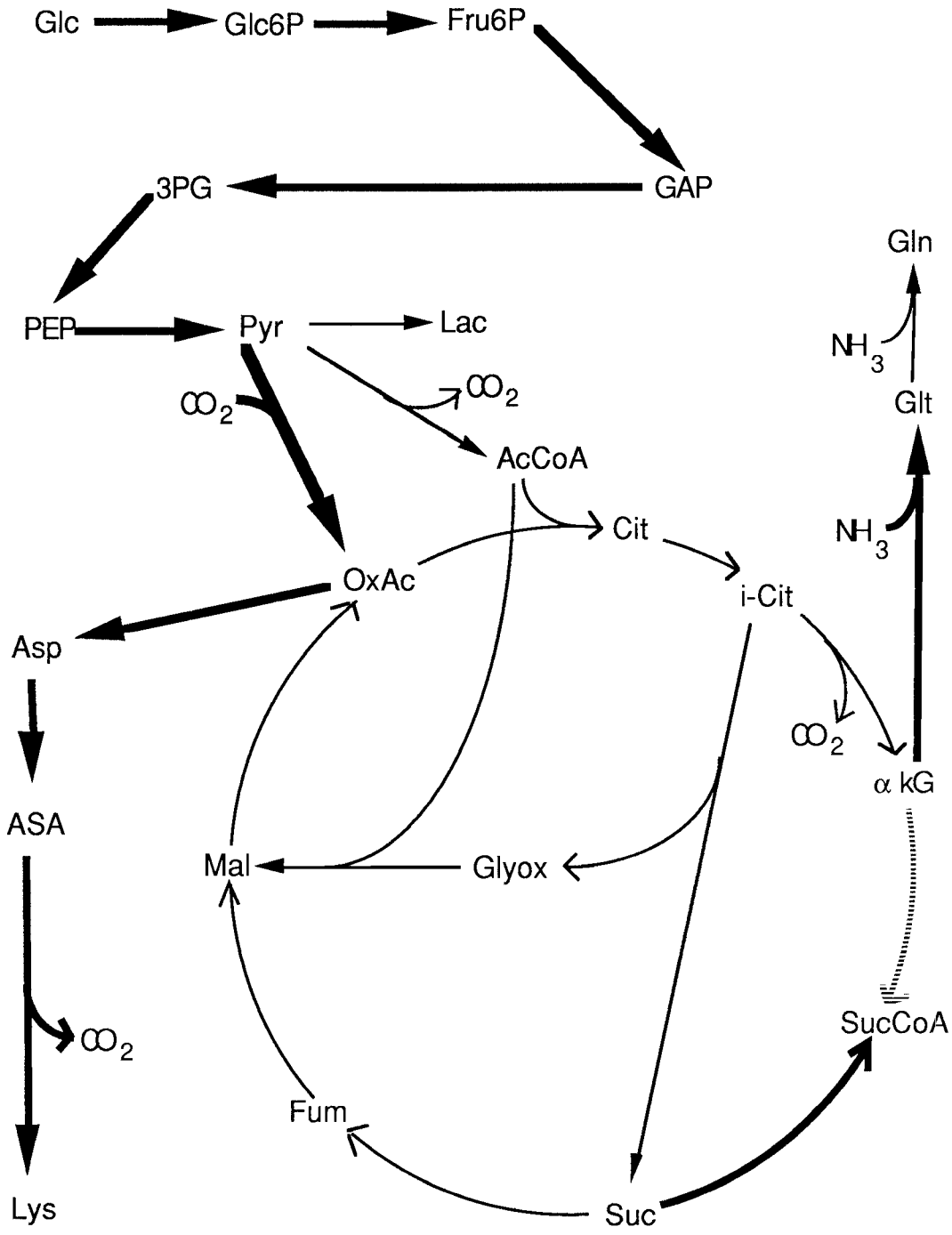


Figure 3: A lysine pathway involving carboxylation of pyruvate.

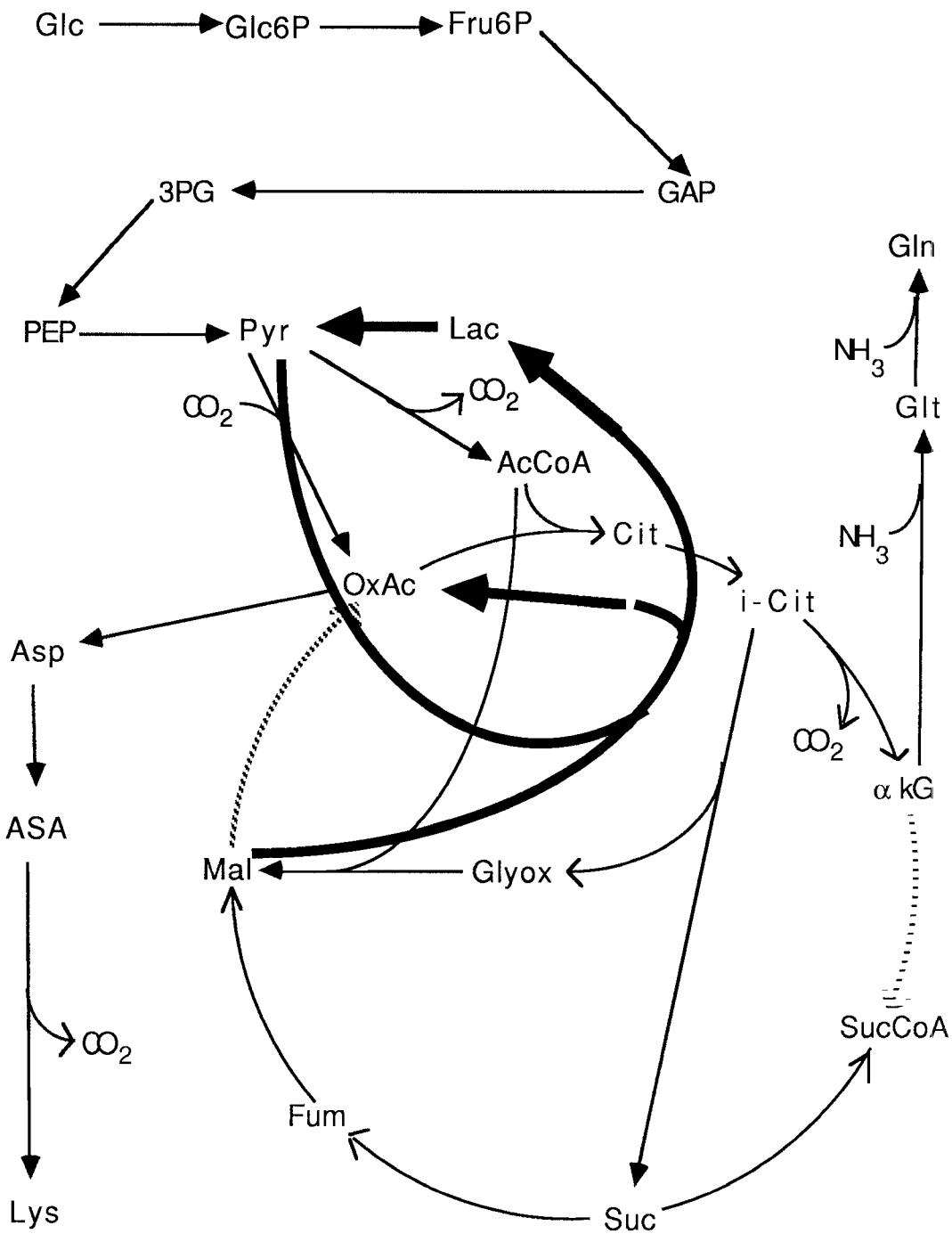


Figure 4: A pathway converting malate to oxaloacetate, with lactate and pyruvate as intermediates

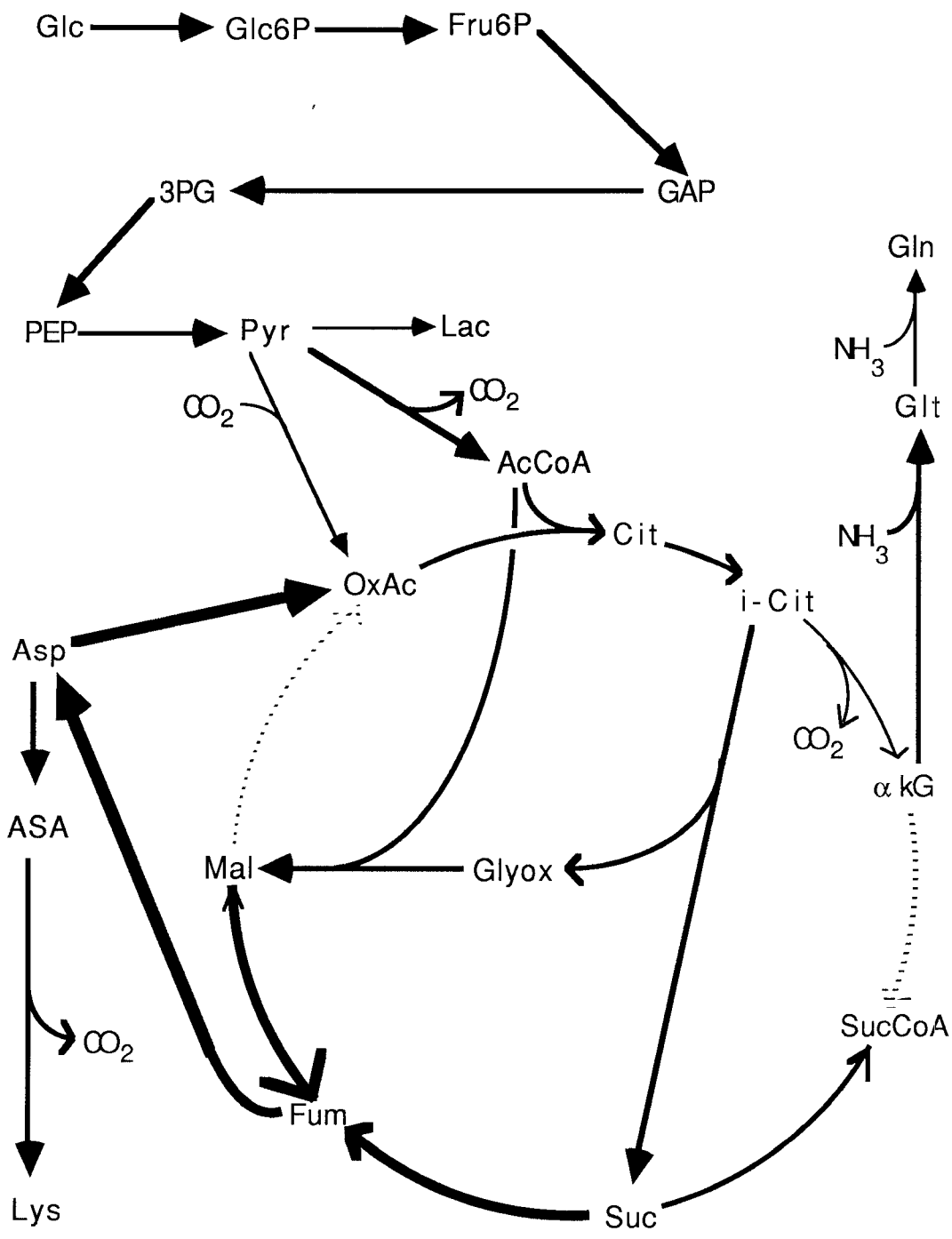


Figure 5: A pathway bypassing malate dehydrogenase by converting fumarate to aspartate.

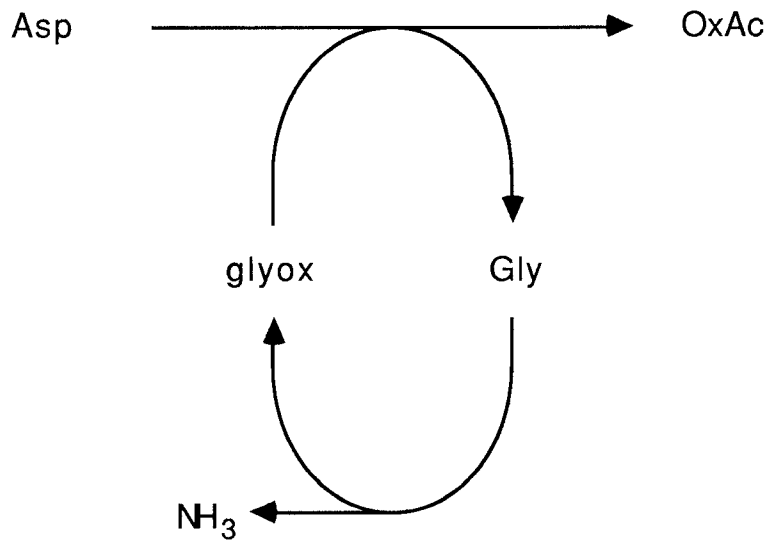


Figure 6: A variation on the conversion of aspartate to oxaloacetate.

The role of oxaloacetate. In the the pathway of Figure 5 and its variant of Figure 6, oxaloacetate is *partly* bypassed, since it is needed only for the synthesis of citrate, and not directly for the synthesis of aspartate and lysine. An interesting question is whether oxaloacetate can be bypassed *altogether* and produce lysine from pyruvate or glucose without the involvement of oxaloacetate at any point. Within the reaction-database used, this turns out to be impossible. Note that no single reaction surrounding oxaloacetate is fixed (present in all pathways); although the particular reactions consuming and producing it vary, the intermediate itself is always present. Thus, oxaloacetate is a key node in the production of lysine.

One might argue that this conclusion is obvious because standard biochemistry textbooks classify lysine in the aspartate family[5-6], and aspartate is commonly synthesized from oxaloacetate. However, the pathways of Figures 1 to 9 involve no less than 9 different bioreactions consuming or producing oxaloacetate; thus, the metabolism in the region of this intermediate can hardly be characterized as fixed. Our conclusion states that any lysine-producing pathway involves at least 2 of these reactions (hence there is no pathway that can avoid the intermediate altogether).

In the pathway of Figure 5 (and its variation suggested by Figure 6) aspartate and lysine are *not* directly derived from oxaloacetate, because fumarate is converted to aspartate by a single enzyme. In fact, aspartate is converted *into* oxaloacetate (rather than the reverse). Thus, the metabolism in the neighborhood of aspartate, fumarate, malate and oxaloacetate is quite different from what one would find in a standard biochemistry textbook. This portion of the metabolism suggests that it *is* possible to derive aspartate without the intervention of oxaloacetate. It turns out however that, within the enzyme database used here, the necessary TCA intermediates (malate or succinate) cannot be produced from glucose without the intervention of oxaloacetate; this constraint necessitates the presence of oxaloacetate in any pathway leading from glucose to lysine. In effect, the real obstacle is that production of fumarate from glucose requires the TCA cycle and hence oxaloacetate.

To illustrate this point better, assume that (in addition to glucose) we could use succinate as an allowed reactant. *A priori* biosynthetic classifications would still entail oxaloacetate as a required intermediate. Inspection of the the pathway of Figure 5 reveals, however, that succinate can be converted to fumarate and on to aspartate (by *aspartate aminolyase*), without the intervention of malate or oxaloacetate. Thus, with succinate as an additional substrate, it is entirely possible to synthesize lysine with a pathway that does not entail oxaloacetate.

If one rests with the preconceived pathways of biochemistry textbooks, one would draw a variety of conclusions about essential enzymes and intermediates. It would, for example, appear safe to assume that the carboxylation of pyruvate to oxaloacetate must involve either *pyruvate carboxylase* or *oxaloacetate decarboxylase*. This assumption would not be correct, because there are non-obvious alternatives, such as the pathway of Figure 6. The pathways in Figures 7,9, and 10, which will be discussed in more detail below, contain other non-obvious possibilities for different biotransformations.

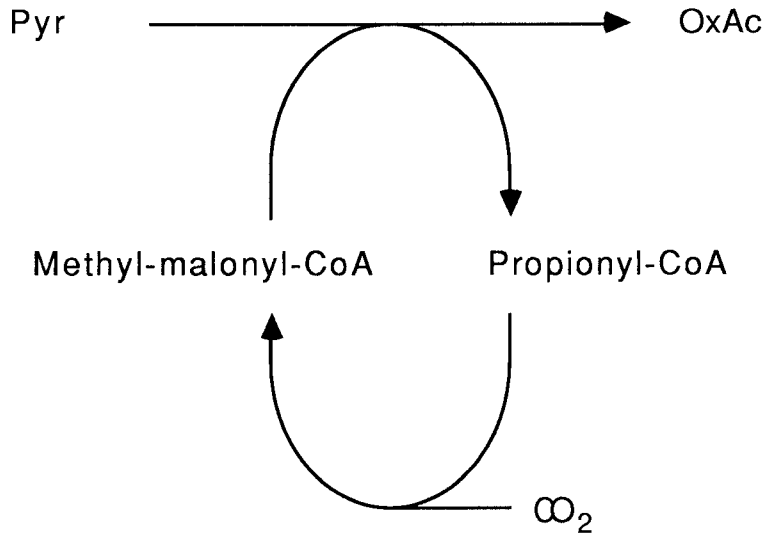


Figure 7: Carboxylation of pyruvate through an alternative pathway, involving Methyl-malonyl-CoA and Propionyl-CoA

Other alternatives. The lysine pathways that have been examined so far involve either *pyruvate dehydrogenase* (the usual entry of pyruvate into the TCA cycle) or *pyruvate carboxylase* (bypasses the TCA cycle altogether). There are, however, pathways which bypass *pyruvate carboxylase* and *pyruvate dehydrogenase*, pointing to other ways in which pyruvate can enter the citric acid cycle. A pathway very similar to the pathway of Figure 3 achieves the carboxylation of pyruvate through *Methyl-malonyl-CoA carboxytransferase* and *Propionyl-CoA carboxylase* (Figure 7). In another pathway similar to Figure 3, direct carboxylation of phosphoenolpyruvate (Figure 8) allows glycolysis to be connected to the TCA cycle through a route that bypasses pyruvate altogether.

A whole set of alternative pathways for the entry of pyruvate in the TCA cycle involves the use of acetate as an intermediate (Figure 9). There are 2 short pathways for the production of acetate from pyruvate, and 3 short pathways for the conversion of acetate to acetyl-CoA or citrate. In this count, bioreactions that have essentially the same stoichiometries are not included. For example, there are two possible bioreactions that can convert oxaloacetate and acetyl-CoA into citrate (namely, *ATP-citrate-lyase* and *Citrate-synthetase*), but only one of them is counted here. By forming combinations that include exactly one pathway producing acetate and one pathway consuming it, a total of 6 pathways can be constructed to convert pyruvate and oxaloacetate into citrate. The presence of an intermediate (here, acetate) produced and consumed in many ways, generally creates an explosion in the number of pathways produced by the methodology.

Of all the alternative pathways produced by the synthesis algorithm only some of the simplest ones were discussed above. It should be remembered, however, that the algorithm can find *all* pathways, and that some very *complex* pathways will be among them. As an example, the simple task of conversion of PEP to pyruvate (which can be achieved in one step by *pyruvate kinase*) is considered

here. The algorithm produced several pathways for this conversion, the longest of which is depicted in Figure 10. The fact that a pathway with so many loops can be successfully constructed is strong evidence that supports the completeness of the algorithm.

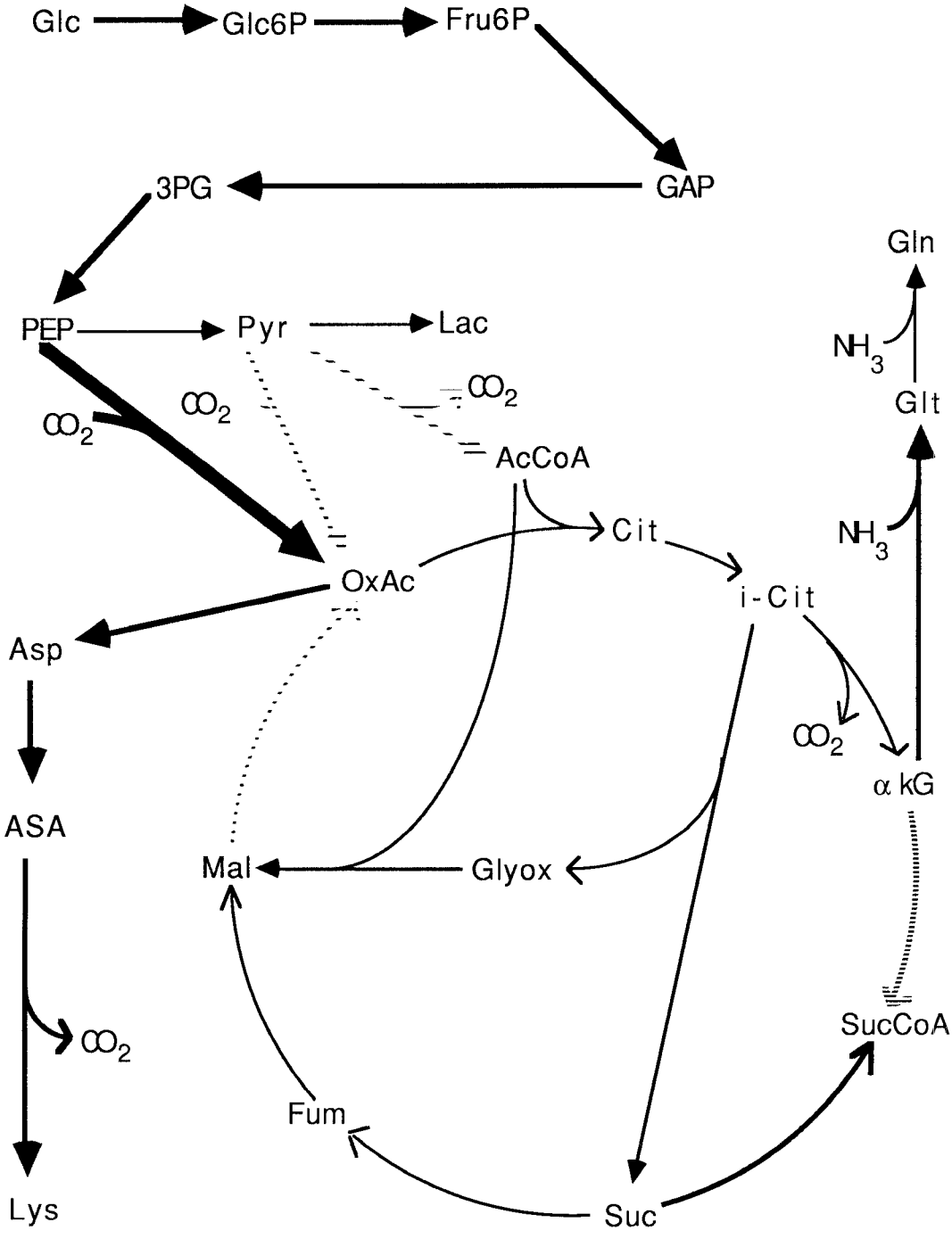


Figure 8: A pathway that carboxylates PEP, bypassing pyruvate

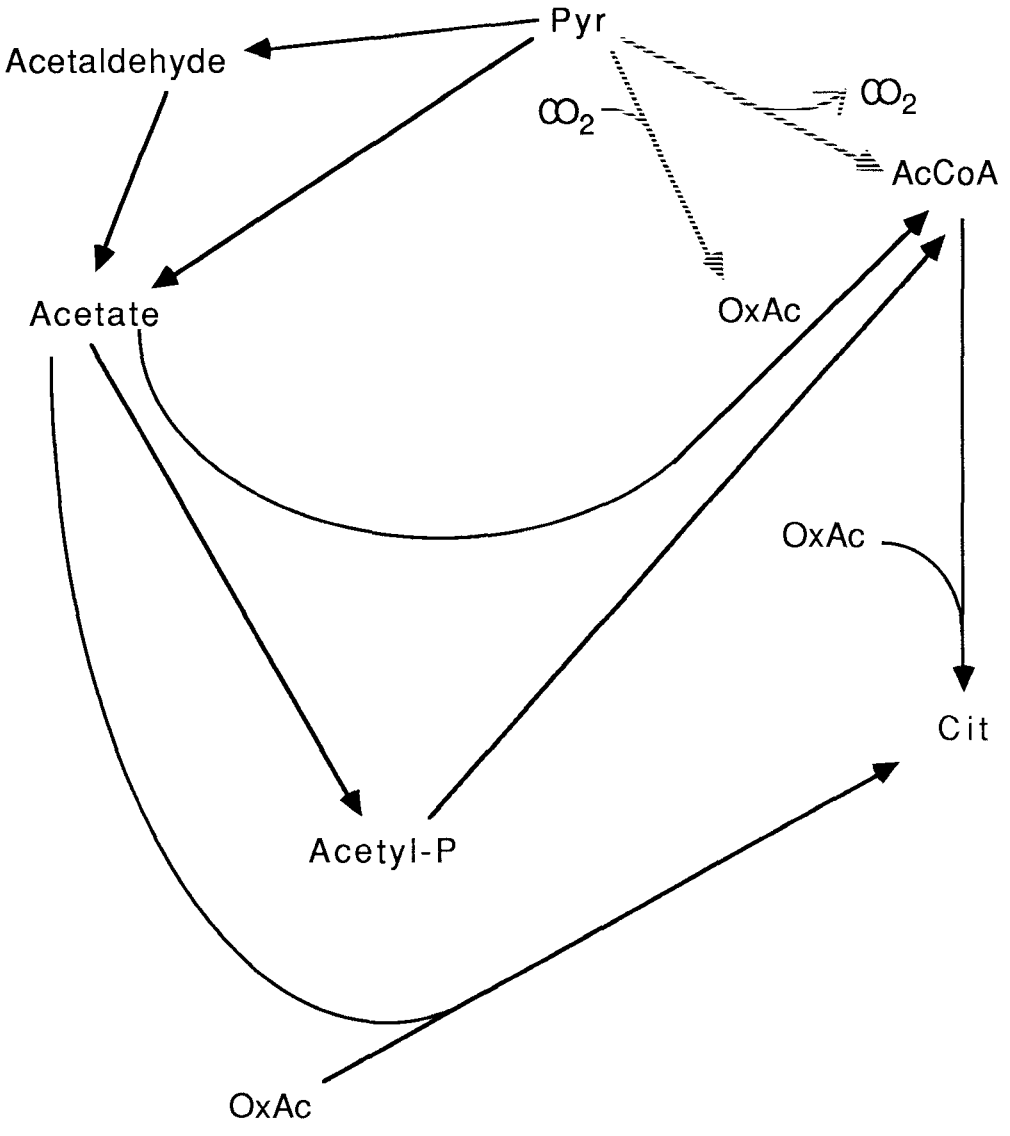


Figure 9: A set of pathways that bypass pyruvate dehydrogenase by converting pyruvate to acetate and then to Acetyl-CoA or citrate

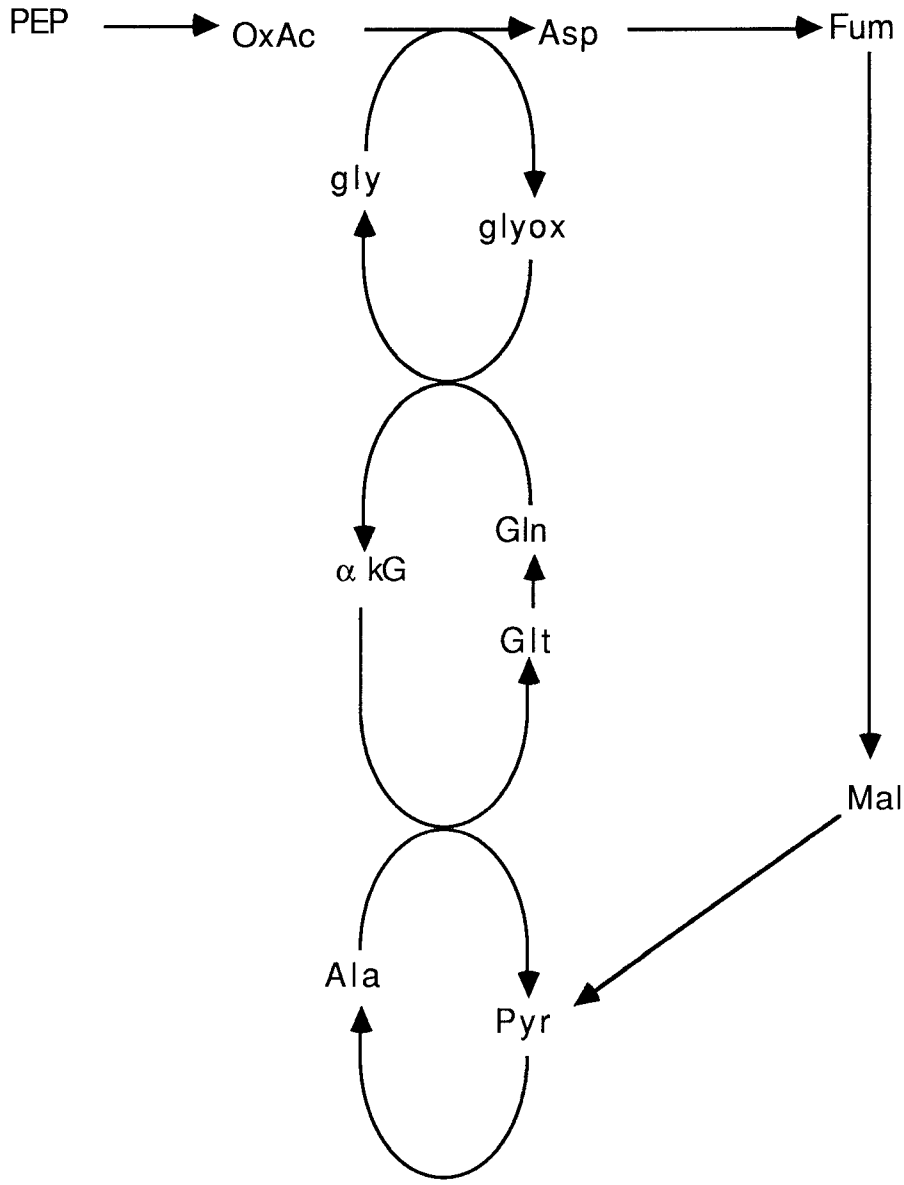


Figure 10: The longest pathway converting PEP to pyruvate

Restrictions on the maximum yield. Some of the most interesting results of applying the synthesis algorithm involve not particular pathways found, but rather demonstrations that no pathways exist to meet certain sets of specifications. As mentioned earlier, the algorithm showed that no pathway can reach lysine from glucose without using oxaloacetate as an intermediate. The algorithm also revealed that the maximum yield of the pathway can exceed 0.67 moles of lysine per mole of glucose only through recovery of carbon dioxide by some bioreaction. In effect, if reactions that consume carbon dioxide are eliminated, the yield is restricted to be 0.67 or less.

One may argue that calculations of the maximum yield can be performed without the algorithm presented here. Such calculations, however, require knowledge of the specific pathway used (or at least the basic characteristics of the pathway, with respect to carbon utilization). Without the systematic examination of all possible pathways, one cannot be certain that other pathways would not

violate the calculations. For example, if some pathway were devised to convert 2 moles of pyruvate to 3 moles of AcCoA (without production and consumption of CO_2):

$$2\ CH_3COCOO^{1-} + 3\ HS\text{-}CoA + 4\ H^{1+} + 2\ NADH \rightarrow 3CH_3CO\text{-}S\text{-}CoA + 3\ H_2O + 2\ NAD^{1+}$$

a yield of lysine over glucose higher than 67% would be achieved – without a reaction recovering CO_2. Given that pyruvate and oxaloacetate participate in a large number of bioreactions, one cannot *a priori* rule out the existence of such a pathway. The algorithm presented here, however, shows that, within the enzyme database used, no such pathway exists.

The satisfaction of the atom and reduction balances imposes, in general, a theoretical maximum yield which is independent of the pathways used. The maximum can be derived by writing down the stoichiometric transformation:

$$C_6H_{12}C_6 + y\ O_2 + z\ NH_3 \rightarrow x\ C_6H_{14}O_2N_2 + b\ H_2O + c\ CO_2$$

and determining the stoichiometric coefficients through atom balances for C, H, N, and O. The coefficient of glucose has already been set to 1, so that the coefficient x is the molar yield of lysine over glucose. Since there are five variables and only four equations, the coefficients y, z, b, c, are determined as functions of the yield x:

$$C_6H_{12}C_6 + (6\text{-}7x)\ O_2 + 2x\ NH_3 \rightarrow x\ C_6H_{14}O_2N_2 + (6\text{-}4x)\ H_2O + (6\text{-}6x)\ CO_2$$

If x>0.857, the coefficient of oxygen becomes negative, which means that oxygen would be actually a *product* of the pathway. Since this is not acceptable, x is set to 0.857 as the maximum yield, leading to a transformation which neither consumes nor produces oxygen:

$$C_6H_{12}C_6 + 1.714\ NH_3 \rightarrow 0.857\ C_6H_{14}O_2N_2 + 2.571\ H_2O + 0.857\ CO_2$$

Thus, arguments that are independent of the actual pathways used can only restrict the maximum yield to 85.7%. Any further restrictions arise either from arguments about the particular pathways involved, or from examining *all* possible pathways, through the algorithm proposed in this paper (since the only alternative algorithm[2] cannot construct *all* pathways).

CONCLUSIONS

The problem of synthesizing biochemical pathways that satisfy linear stoichiometric constraints was discussed in this paper. An algorithm for the solution of the problem was presented, based on the iterative satisfaction of constraints, and the transformation of the initial set of reactions (which can be thought of as one-step pathways) into a final set of pathways which satisfy all constraints. The algorithm is correct and complete and has satisfactory computational performance, considering the inherent computational complexity of the problem (Appendix C). More details on the operation of the algorithm are given in Appendices A and B.

The algorithm is of significant value in the investigation of alternative biochemical pathways to achieve a given biotransformation (which is defined by a set of stoichiometric specifications). It can also produce pathways that bypass bottlenecks of a given pathway. A variety of alternative non-obvious routes for the synthesis of lysine demonstrates the utility of computer-based, systematic construction of pathways. Furthermore, the algorithm can identify fundamental limitations that govern the biochemical pathways and the process. In the case of lysine-producing pathways, it was shown that oxaloacetate is always present as an intermediate, and that in the absence of recovery of carbon dioxide by some bioreaction the yield of lysine over glucose is restricted to be 0.67 or less.

NOMENCLATURE

Abbreviations for metabolic intermediates used in bioreaction networks

ABBREVIATION	METABOLITE
Glc	Glucose
Glc6P	Glucose-6-phosphate
Fru6P	Fructose-6-phosphate
GAP	Glyceraldehyde-3-phosphate
3PG	3-phosphoglycerate
2PG	2-phosphoglycerate
PEP	Phosphoenolpyruvate
Pyr	Pyruvate
AcCoA (or Acetyl-CoA)	Acetyl-Coenzyme-A
Cit	Citrate
i-Cit	Isocitrate
OxAc	Oxaloacetate
Glyox	Glyoxylate
Fum	Fumarate
Mal	Malate
Suc	Succinate
SucCoA	Succinyl-Coenzyme-A
αkG	α-ketoglutarate
Glt or Glu	Glutamate
Gln	Glutamine
Asp	Aspartate
ASA	Aspartate-semialdehyde
Lys	Lysine
Gly	Glycine
Ala	Alanine

REFERENCES

1. Mavrovouniotis, M. L. "Computer-Aided Design of Biochemical Pathways" Ph.D. Thesis, Massachusetts Institute of Technology, 1989.

2. Seressiotis, A., and Bailey, J. E. "MPS: An Artificially Intelligent Software System for the Analysis and Synthesis of Metabolic Pathways" *Biotechnology and Bioengineering*, 31:587-602, 1988.

3. Mostow, J. "Rutgers Workshop on Knowledge-Based Design" *SIGART Newsletter* (90):19-32, October, 1984.

4. Mostow, J. "Toward Better Models of the Design Process" *The AI Magazine* 6(1):44-56, Spring, 1985.

5. Rawn, J. D. *Biochemistry*, pp. 883-888. Harper and Row, New York, 1983.

6. Mandelstam, J., McQuillen, K., and Dawes, I. *Biochemistry of Bacterial Growth, 3rd edition*, pp. 163-165. Wiley, New York, 1982.

APPENDIX A: DESCRIPTION OF THE ALGORITHM

Given a set of stoichiometric requirements and a database of biochemical reactions (i.e., reaction stoichiometries), the developed algorithm synthesizes all biochemical pathways that satisfy the requirements. An informal description of the workings of the algorithm was provided in the body of the paper. In this Appendix, the workings of the algorithm are described more rigorously. Appendix B provides an example of a step-by-step application of the algorithm. Appendix C briefly discusses theoretical and practical issues regarding the computational complexity of the problem.

Reaction-Processing Phase. In order to account for the reversibility of reactions, each thermodynamically reversible reaction is decomposed into a forward and a backward reaction. From this point on, we prohibit the participation of both the forward and the reverse reaction in the same pathway, because such a pathway would be redundant.

The constraint placed on the original reaction is transformed into constraints on these two portions. There are only three possible specifications on reaction coefficients. For a reaction R_k, and its coefficient a_k:

- R_k may occur in the pathway, i.e., $a_k \geq 0$.
- R_k must occur in the pathway, i.e., $a_k > 0$.
- R_k must not occur in the pathway, i.e., $a_k = 0$.

From the previous constraints, the first category (allowed reactions) can be simply ignored. The third category (excluded reactions) can be satisfied right from the start, by removing such reactions from the database of available reactions. The constraints referring to required reactions, on the other hand, will only be satisfied in the last phase of the algorithm.

Metabolite-Processing Phase. At each subsequent stage in the synthesis of algorithm, the problem state is characterized by a set of partial pathways. From these, one can construct new pathway combinations to satisfy the remaining requirements. Each partial solution, then, consists of the following elements:

- The set of constraints that have not yet been satisfied, i.e., the set of metabolites that still remain to be processed.
- The set of active, incomplete pathways constructed so far. These are pathways that satisfy the already-processed constraints. The initial set of reactions is the starting set of (one-step) pathways, when no constraints have been processed yet.
- Back-pointers which show, for each remaining metabolite, the pathways in which it participates as a reactant, product, or intermediate — three separate lists must be kept.

At each pathway-expansion step, one of the remaining metabolite is chosen to be processed and the requirements on that metabolite become the current goal. The most suitable metabolite is the one which participates (as a reactant or product) in the smallest possible number of pathways that are active in the current state of the problem. The algorithm finds a modification of the set of pathways such that all surviving pathways satisfy the requirement. This involves the construction of new pathways as linear combinations of existing ones, as well as deletion of pathways from the working set. More specifically, for S the metabolite being processed and using the backward-pointers readily available in each metabolite, two subsets of the current pathway set, L, are assembled:

- The list of pathways that produce the metabolite: $L_p=\{P_i|$ S participates in P_i with a net stoichiometric coefficient $a_i>0\}$.
- The list of pathways that consume the metabolite: $L_c=\{P_i|$ S participates in P_i with a net stoichiometric coefficient $a_i<0\}$.
- The list of pathways in which the metabolite participates as an intermediate: $L_r=\{P_i|$ S participates in some reaction R with coefficient $r_i\neq0$, but S does not participate in the net transformation of P_i, i.e., $a_i=0\}$.
- The list of pathways in which the metabolite does not participate at all $L_n=\{P_i|$ the coefficient of S in each reaction R of P_i is $r_i=0\}$.

The pathways that may, at this step of the algorithm, be deleted from the current set will be pathways from the lists L_p, L_c, and L_r, depending on the nature of the constraint. The pathways that may be constructed are linear combinations using exactly one pathway from L_c and exactly one pathway from L_p:

- Combination pathways: $L_e=\{a_kP_i-a_iP_k \mid P_i\in L_c;\ P_k\in L_p;\ P_i$ and P_k do not involve the same reaction in different directions; and a_i and a_k are the net coefficients with which S participates in P_i and $P_k\}$. Since $P_i\in L_c$, $a_i<0$ and the combination $a_kP_i-a_iP_k$ has positive coefficients; thus, it is a legitimate combination of pathways. The net coefficient of S in $a_kP_i-a_iP_k$ is $a_ka_i-a_ia_k=0$. Thus, for all pathways in L_e, S is only an intermediate; it is neither a net reactant nor a net product. As was noted earlier, we exclude combinations of two pathways that involve the same reaction in different directions.

For constraints on reactants and products, the construction of the new set of active pathways is delineated below. The different cases are listed by priority, i.e., in the order in which they should be applied. Once a particular case applies then the remaining cases are automatically excluded*.

- If S is an excluded product and a required reactant (i.e., $a_k < 0$), all combination pathways are constructed, and the producing pathways are deleted. In effect: $L\leftarrow L\cup L_e-L_p$ (i.e., the new set L is obtained as the old L plus L_e minus L_p). This constraint will receive additional attention in the pathway-marking phase.
- If S is an excluded reactant and a required product (i.e., $a_k > 0$), then: $L\leftarrow L\cup L_e-L_c$. This constraint will receive additional attention in the pathway-marking phase.
- If S is an excluded product and an allowed reactant (i.e., $a_k \le 0$), then: $L\leftarrow L\cup L_e-L_p$
- If S is an excluded reactant and an allowed product (i.e., $a_k \ge 0$), then: $L\leftarrow L\cup L_e-L_c$
- If S is an excluded reactant and an excluded product (i.e., $a_k = 0$), then: $L\leftarrow L\cup L_e-L_c-L_p$
- If S is an excluded intermediate, then: $L\leftarrow L-L_c-L_p-L_r$, or, equivalently, $L\leftarrow L_n$
- If S is a required intermediate, then: $L\leftarrow L$. This constraint will receive additional attention in the pathway-marking phase.
- If S is an allowed reactant, an allowed product, and an allowed intermediate, then no true constraint exists. Thus, the set of active pathways is carried intact to the next iteration: $L\leftarrow L$
- If S is a required intermediate , then: $L\leftarrow L$. This constraint will receive additional attention in the pathway-marking phase.

Appendix B provides an example of a step-by-step application of the algorithm, illustrating the construction of combination pathways and the deletion of existing pathways. After the processing of the constraint, there is a new set of active pathways which satisfy the constraint, with the exception that for strict-inequality constraints, i.e., required products ($a_k > 0$), required reactants ($a_k < 0$), and required intermediates, only the corresponding loose-inequality constraints are guaranteed to be satisfied. The strict-inequality constraints will receive additional consideration in the last phase of the algorithm.

In addition to the set of active pathways, L, the whole *state of the design* that was described earlier must also be properly updated after each constraint is processed. For example, to update the back-pointers that point from each metabolite to the pathways in which it participates, pointers corresponding to deleted pathways must be removed and pointers corresponding to new pathways must be added.

Pathway-Marking Phase. At the end of the metabolite-processing phase, there is a final set of active pathways satisfying all of the requirements, except the strict-inequality constraints for which

---

* The constraints are assumed to be consistent. For example, if S is a required product, it cannot be a required reactant.

only the corresponding loose inequalities are satisfied. Because of the linear nature of the requirements, all combinations of pathways also satisfy the constraints. Some pathways from the final set also satisfy a subset of original strict-inequality constraints. Because the requirements are linear, combinations of pathways satisfy the *union* of the constrains satisfied by their constituent pathways. Thus, the pathways satisfying the original stoichiometric constraints of the synthesis problem are all those combinations of pathways from the final set which have at least one constituent pathway satisfying each of the strict-inequality constraints. In other words, the combination must include:

- ◊ at least one pathway consuming each required reactant,
- ◊ at least one pathway producing each required product, and
- ◊ at least one pathway containing each required intermediate.
- ◊ at least one pathway in which each required reaction participates.

Properties of the Algorithm. If a combination of pathways from the final set (produced by the synthesis algorithm) contains at least one constituent pathway satisfying each of the strict inequality requirements (referring to required reactants, products, intermediates, or reactions), then the combination pathway satisfies all of the initial stoichiometric requirements. Thus, the algorithm generates only correct solutions to the synthesis problem. The proof[1] is based on the fact that each of the original requirements is satisfied in one of the three phases, and after each constraint is satisfied it cannot be subsequently violated.

The synthesis algorithm creates a final set of pathways such that: Any pathway satisfying the original stoichiometry constraints is a combination of pathways from the final set, with one constituent pathway satisfying each strict-inequality constraint. Thus, the algorithm is complete. The proof[1] is based on the fact that each metabolite-processing step preserves all those pathways or combinations that may lead, in the end, to feasible solutions.

When the algorithm is not permitted to run to completion (because of limited computational resources), it can provide useful *partial* results. Specifically, it will return a list of pathways that satisfy only *some* of the constraints involved; it will also return the list of *unprocessed constraints*. For example, in the example of Appendix B, if the algorithm must stop before the last step, it returns a list of four pathways that satisfy all constraints *except for the constraint designating E as an excluded reactant and excluded product;* the algorithm also indicates that the constraint on E has not been satisfied.

Appendix B provides an example of a step-by-step application of the algorithm. Appendix C briefly discusses theoretical and practical issues regarding the computational complexity of the problem.

APPENDIX B: A DETAILED EXAMPLE OF THE OPERATION OF THE ALGORITHM

A step-by-step application of the algorithm for a synthesis problem, suggested as a challenge by the reviewers of this paper, is presented here. The set of reactions under consideration is:

a. $A \rightarrow B$

b. $B \leftrightarrow C$

c. $C \leftrightarrow D$

d. $C + D \leftrightarrow F + K$

e. $F + K \leftrightarrow H + E$

f. $H + D \leftrightarrow E + F$

g. $A \leftrightarrow E$

h. $E \rightarrow F + G$

k. $F \leftrightarrow G$

m. $G \rightarrow L$

All metabolites are designated as excluded reactants and excluded products, with the exception of A, which is a required reactant, and L, which is a required product.

We first construct the reverse reactions, for reactions b, c, d, e, f, g, and k. We designate the reverse reactions as -b, -c, -d, -e, -f, -g, and -k respectively. Each (forward or reverse) reaction is now considered a one-step partial pathway.

To complete the description of the initial state of the problem, we also list separately each metabolite and the pathways in which it participates. Two systems of notation are used here for pathways. Representing only the reactions from which a pathway is constructed, an expression like [2a, 2-g, b] denotes a pathway that is constructed as a linear combination of the reactions a, -g, and b, with coefficients 2, 2, and 1, respectively. To represent instead the overall transformation accomplished by this pathway, the expression $2E \rightarrow \rightarrow B + C$ is used. The two alternative expressions can be combined into $2E \rightarrow[2a, 2\text{-}g, b]\rightarrow B + C$. Using this notation the initial state of the problem entails the pathways:

$A \rightarrow[a]\rightarrow B$

$B \rightarrow[b]\rightarrow C$

$C \rightarrow[-b]\rightarrow B$

$C \rightarrow[c]\rightarrow D$

$D \rightarrow[-c]\rightarrow C$

$C + D \rightarrow[d]\rightarrow F + K$

$F + K \rightarrow[-d]\rightarrow C + D$

$F + K \rightarrow[e]\rightarrow H + E$

$H + E \rightarrow[-e]\rightarrow F + K$

$H + D \rightarrow[f]\rightarrow E + F$

$E + F \rightarrow[-f]\rightarrow H + D$

$A \rightarrow[g]\rightarrow E$

$E \rightarrow[-g]\rightarrow A$

$E \rightarrow[h]\rightarrow F + G$

$F \rightarrow[k]\rightarrow G$

$G \rightarrow[-k]\rightarrow F$

$G \rightarrow[m]\rightarrow L$

The set of metabolites with the pathways in which they participate is:

A: [a], [g], [-g]
B: [b], [-b], [a]
C: [b], [-b], [c], [-c], [d], [-d]
D: [c], [-c], [d], [-d], [f], [-f]
E: [e], [-e], [f], [-f], [g], [-g], [h]
F: [d], [-d], [e], [-e], [f], [-f], [h], [k], [-k]
G: [h], [k], [-k], [m]
H: [e], [-e], [f], [-f]
K: [d], [-d], [e], [-e]
L: [m]

Following the algorithm, the metabolites that participate in fewer pathways must be processed first. L, which participates in only one pathway, is a required product (and an excluded reactant). Since L is produced by one partial pathway and is not consumed by any partial pathway, processing the constraints on this metabolite does not change any pathway.

The next metabolite that is processed must be either A or B; the order in which these two metabolites are processed does not affect the results and we arbitrarily choose B. One new pathway are constructed: [a,b] as a combination of [a] and [b]; this operation is denoted as [a]+[b]=[a,b]. Note that it is not permissible to construct the pathway [b]+[-b], because it would involve the same reaction in both the forward and reverse directions; such a pathway would be a trivial loop, not accomplishing any net transformation. After the combination [a,b] is constructed, the pathways [a], [b], and [-b] are deleted. The set of active pathways is now:

$A \rightarrow[a, b]\rightarrow C$

$C \rightarrow[c]\rightarrow D$

$D \rightarrow[-c]\rightarrow C$

$C + D \rightarrow[d]\rightarrow F + K$

$F + K \rightarrow[-d]\rightarrow C + D$

$F + K \rightarrow[e]\rightarrow H + E$

$H + E \rightarrow[-e]\rightarrow F + K$

$H + D \rightarrow[f]\rightarrow E + F$

E + F →[-f]→ H + D

A →[g]→ E

E →[-g]→ A

E →[h]→ F + G

F →[k]→ G

G →[-k]→ F

G →[m]→ L

The updated set of metabolites that still need to be processed becomes:

A: [a, b], [g], [-g]
C: [a, b], [c], [-c], [d], [-d]
D: [c], [-c], [d], [-d], [f], [-f]
E: [e], [-e], [f], [-f], [g], [-g], [h]
F: [d], [-d], [e], [-e], [f], [-f], [h], [k], [-k]
G: [h], [k], [-k], [m]
H: [e], [-e], [f], [-f]
K: [d], [-d], [e], [-e]

The metabolite A is processed next. Since A is a required reactant and excluded product, a new combination pathway are constructed as [-g]+[a,b]=[-g,a,b], and only pathway [-g] is deleted. For the next step G is selected arbitrarily among the metabolites G, H, and K (which participate in the same number of reactions). In processing G, there are two pathways consuming it ([-k] and [m]) and two pathways producing ([h] and [k]). Hence, four combinations would be constructed, except that [k] cannot be combined with [-k]. Three legitimate combinations remain, namely: [h]+[-k]=[h, -k]; [h]+[m]=[h, m]; [k]+[m]=[k, m]. The original four pathways in which G participated are deleted.

After the processing of A and G, as described above, the active pathways are:

A →[a, b]→ C

C →[c]→ D

D →[-c]→ C

C + D →[d]→ F + K

F + K →[-d]→ C + D

F + K →[e]→ H + E

H + E →[-e]→ F + K

H + D →[f]→ E + F

E + F →[-f]→ H + D

A →[g]→ E

E →[-g, a, b]→ C

E →[h, -k]→ 2 F

E →[h, m]→ F + L

F →[k, m]→ L

The updated set of metabolites that still need to be processed becomes:

C: [a, b], [c], [-c], [d], [-d], [-g, a, b]
D: [c], [-c], [d], [-d], [f], [-f]
E: [e], [-e], [f], [-f], [g], [-g, a, b], [h, m], [h, -k]
F: [d], [-d], [e], [-e], [f], [-f], [h, m], [k, m], [h, -k]
H: [e], [-e], [f], [-f]
K: [d], [-d], [e], [-e]

The metabolite K, participating in four pathways, is processed next. The combinations [d]+[e]=[d, e], and [-e]+[-d]=[-e, -d] are created, and the pathways [d], [-d], [e], and [-e] are deleted. The set of active pathways becomes:

A →[a, b]→ C

C →[c]→ D

D →[-c]→ C

C + D →[d, e]→ H + E

H + E →[-e, -d]→ C + D

H + D →[f]→ E + F

E + F →[-f]→ H + D

A →[g]→ E

E →[-g, a, b]→ C

E →[h, -k]→ 2 F

E →[h, m]→ F + L

F →[k, m]→ L

The updated set of metabolites that still need to be processed becomes:

C: [a, b], [c], [-c], [d, e], [-e, -d], [-g, a, b]
D: [c], [-c], [d, e], [-e, -d], [f], [-f]
E: [d, e], [-e, -d], [f], [-f], [g], [-g, a, b], [h, m], [h, -k]
F: [f], [-f], [h, m], [k, m], [h, -k]
H: [d, e], [-e, -d], [f], [-f]

Processing H in a very similar fashion, two combination pathways are constructed, namely [-f]+[-e,-d]=[-f,-e,-d] and [d,e]+[f]=[d,e,f]; it should be mentioned again that pathways that involve the same reaction in both the forward and the reverse reaction are never constructed. The pathways now become:

A →[a, b]→ C

C →[c]→ D

D →[-c]→ C

$C + 2\,D \rightarrow [d, e, f] \rightarrow 2\,E + F$

$2\,E + F \rightarrow [-f, -e, -d] \rightarrow C + 2\,D$

$A \rightarrow [g] \rightarrow E$

$E \rightarrow [-g, a, b] \rightarrow C$

$E \rightarrow [h, -k] \rightarrow 2\,F$

$E \rightarrow [h, m] \rightarrow F + L$

$F \rightarrow [k, m] \rightarrow L$

The updated set of metabolites that still need to be processed becomes:

C: [a, b], [c], [-c], [d, e, f], [-f, -e, -d], [-g, a, b]
D: [c], [-c], [d, e, f], [-f, -e, -d]
E: [d, e, f], [-f, -e, -d], [g], [-g, a, b], [h, m], [h, -k]
F: [d, e, f], [-f, -e, -d], [h, m], [k, m], [h, -k]

Since D involves now only 4 pathways, it is processed next. The fact that the coefficient of D in [d, e, f] and [-f, -e, -d] is 2 must be reflected in the construction of the combinations. The new pathways are constructed as 2[c]+[d,e,f]=[2c, d, e, f] and [-f,-e,-d]+2[-c]=[-f, -e, -d, 2 -c], and all four pathways that involved D are deleted. The set of active pathways is now significantly smaller:

$A \rightarrow [a, b] \rightarrow C$

$3\,C \rightarrow [2c, d, e, f] \rightarrow 2\,E + F$

$2\,E + F \rightarrow [-f, -e, -d, 2\,-c] \rightarrow 3\,C$

$A \rightarrow [g] \rightarrow E$

$E \rightarrow [-g, a, b] \rightarrow C$

$E \rightarrow [h, -k] \rightarrow 2\,F$

$E \rightarrow [h, m] \rightarrow F + L$

$F \rightarrow [k, m] \rightarrow L$

Only three metabolites still need to be processed:

C: [a, b], [2c, d, e, f], [-f, -e, -d, 2 -c], [-g, a, b]
E: [2c, d, e, f], [-f, -e, -d, 2 -c], [g], [-g, a, b], [h, m]
F: [2c, d, e, f], [-f, -e, -d, 2 -c], [h, m], [k, m], [h, -k]

As it participates in only four pathways, C is processed next and leads to two combinations, 3[a,b]+[2c,d,e,f]=[3a, 3b, 2c, d, e, f] and 3[-g, a,b]+[2c,d,e,f]=[3 -g, 3a, 3b, 2c, d, e, f]. After deletion the original four pathways, the active pathways are:

$3\,A \rightarrow [3a, 3b, 2c, d, e, f] \rightarrow 2\,E + F$

$A \rightarrow [g] \rightarrow E$

$E \rightarrow [3\,-g, 3a, 3b, 2c, d, e, f] \rightarrow F$

$E \rightarrow [h, -k] \rightarrow 2\,F$

$E \rightarrow [h, m] \rightarrow F + L$

$$F \rightarrow [k, m] \rightarrow L$$

The two metabolites yet to be processed are:

E: [3a, 3b, 2c, d, e, f], [g], [3 -g, 3a, 3b, 2c, d, e, f], [h, m], [h, -k]

F: [3a, 3b, 2c, d, e, f], [3 -g, 3a, 3b, 2c, d, e, f], [h, m], [k, m], [h, -k]

The two metabolites participate in the same number of pathways, and can be processed in either order to yield the final results. Processing F first leads to three new combinations of pathways: [3a, 3b, 2c, d, e, f]+[k, m]=[3a, 3b, 2c, d, e, f, k, m]; [h, m] + [k, m] = [h, k, 2 m]; and finally [3 -g, 3a, 3b, 2c, d, e, f]+[k, m]=[3 -g, 3a, 3b, 2c, d, e, f, k, m]. After the original 5 pathways in which F participated are deleted, the remaining active pathways are:

$$3\ A \rightarrow [3a, 3b, 2c, d, e, f, k, m] \rightarrow 2\ E + L$$

$$A \rightarrow [g] \rightarrow E$$

$$E \rightarrow [3\text{ -g}, 3a, 3b, 2c, d, e, f, k, m] \rightarrow L$$

$$E \rightarrow [h, k, 2m] \rightarrow 2\ L$$

The remaining unprocessed metabolite, E, participates in all four pathways. Processing E (and omitting pathways that include the same reaction in opposing directions) leads to the combinations: $\frac{1}{3}$ [3a, 3b, 2c, d, e, f, k, m] + $\frac{2}{3}$ [3 -g, 3a, 3b, 2c, d, e, f, k, m] = [2 -g, 3a, 3b, 2c, d, e, f, k, m]*; [3a, 3b, 2c, d, e, f, k, m] + 2[h, k, 2m] = [3a, 3b, 2c, d, e, f, 3k, 5m, 2h]; and the much simpler [g]+[h, k, 2m]=[g, h, k, 2m]. Thus, the final pathways are:

$$A \rightarrow [2\text{ -g}, 3a, 3b, 2c, d, e, f, k, m] \rightarrow L \qquad [P_1]$$

$$3\ A \rightarrow [3a, 3b, 2c, d, e, f, 3k, 5m, 2h] \rightarrow 5\ L \qquad [P_2]$$

$$A \rightarrow [g, h, k, 2m] \rightarrow 2\ L \qquad [P_3]$$

These three pathways are feasible solutions to the original synthesis problem. All other feasible pathways are linear combinations of pathways from this set, with positive coefficients. Other methods for pathway synthesis[2] utilize special rules to produce only the basic pathways; for example, they apply a rule stating that "if an enzyme requires a substrate already used by the pathway to that enzyme, then this substance must be produced by the same set of enzymes used in the pathway". The example in this appendix shows that the method presented here achieves the construction of basic pathways (and avoids the construction of redundant ones) without additional rules of this type.

The algorithm of Seressiotis and Bailey[2] also discusses the issue of genotypical independence of pathways, as contrasted to linear independence. One may observe that of the three pathways constructed only two are linearly independent: pathway $[P_2]$ can be obtained as $[P_1]+2[P_3]$. All three pathways are useful, however, because they are genotypically independent, i.e., they involve independent sets of enzymes: Although linearly $[P_2]=[P_1]+2[P_3]$, the set of enzymes of $[P_2]$ is not the union of the respective sets for $[P_1]$ and $[P_3]$. Specifically, the enzyme g exists in both $[P_1]$ and $[P_3]$ but not in $[P_2]$.

Appendix C briefly discusses theoretical and practical issues regarding the computational complexity of the problem.

---

* To obtain smaller integer coefficients for the combination pathway, the fractions 1/3 and 2/3 were used instead of 1 and 2 in the construction of the combination. This has the same effect as dividing the resulting pathway by 3; clearly, the essence of the transformation and the overall significance of the pathway are not affected by multiplicative constants. Only the molar *proportions* of metabolites and reactions matter.

APPENDIX C: COMPUTATIONAL COMPLEXITY

The number of pathways that satisfy a set of stoichiometric constraints is, in the worst case, exponential in the number of reactions. Consider the reactions depicted in Figure 11. For each diamond (numbered as D_1, D_2, etc.) consisting of two parallel branches, a pathway can follow either the upper branch or the lower branch. If there are n diamonds (and 4n=m reactions), there are n–1 junctions where these choices occur. Thus, there are $2^{n-1}=2^{m/4-1}$ distinct pathways. These are all genotypically independent: Since no two of them involve the same set of choices (at the junctions), it follows that no two of them involve the same set of enzymes.

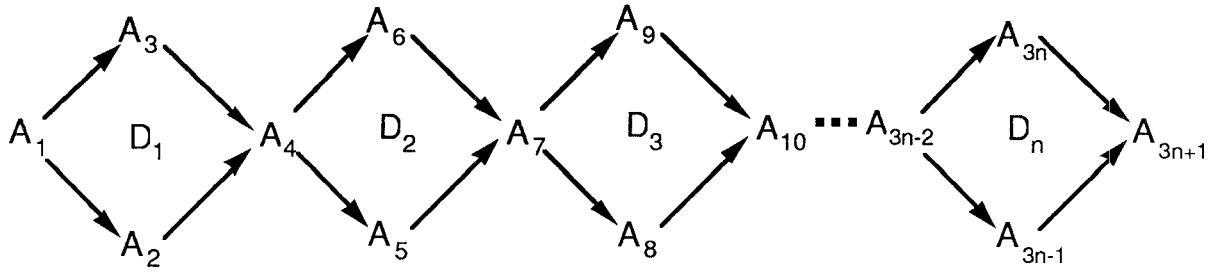


Figure 11: A set of reactions giving rise to an exponential number of pathways

Since the algorithm described here (as well as the previous algorithm of Seressiotis and Bailey[2]) constructs all genotypically independent pathways, the algorithm would require time (and storage space) exponential in the number of reactions. Thus, the algorithm's worst-case complexity is at least exponential.

In practice, however, the metabolism contains long sequences of reactions but few parallel branches of the type of Figure 11. Thus, with careful design of the computer programs it is possible to obtain results more efficiently than the worst-case complexity suggests.

It is useful to discuss, in the context of computational complexity, why the metabolite-processing phase of the pathway does not necessarily start from metabolites that are required reactants (which is the approach followed by other algorithms[2]), and may instead start from other intermediates. In the formulation of the problem in this paper, constraints are imposed on all metabolites. When the algorithm selects the next constraint to satisfy, it picks the one that appears easiest to process (an approach reminiscent of *greedy algorithms*); this would be the metabolite that participates in the smallest number of reactions, regardless of whether the metabolite is a required reactant or an excluded reactant.

The fact that the algorithm processes not only designated required reactants (or products) is an important factor in guaranteeing the completeness of the algorithm and guarding it against computational complexity. Consider the simple pathway of Figure 7, and suppose that the whole reaction database consists of the two reactions in the figure, and the objective is to convert pyruvate to oxaloacetate. The Seressiotis and Bailey algorithm[2] cannot start the construction of a pathway from the reaction:

pyruvate + methyl-malonyl-CoA → oxaloacetate + propionyl-CoA

because this reaction uses methyl-malonyl-CoA, which is not available as a reactant. Likewise, that algorithm can not start from:

propionyl-CoA → methyl-malonyl-CoA + CO_2

because the reaction requires propionyl-CoA which is also not available. The Seressiotis and Bailey algorithm fails to see that, taken as a cluster, these two reactions achieve the desired transformation. The algorithm presented here, on the other hand, considers the constraint that designates propionyl-

CoA as an excluded reactant and excluded product, and immediately constructs the pathway of Figure 7 to satisfy the constraint.

There is also a gain in efficiency through the approach presented here. In the investigation of the production of lysine, the last long section of the pathway (from aspartate to lysine) is unique. Since the Seressiotis and Bailey approach[2] proceeds from reactants to products, it will *reconstruct* that long section of the pathway for every single alternative pathway that leads to aspartate. With the algorithm, the section from aspartate to lysine is processed very early, because each intermediate in that section participates in only two reactions. Once that section is constructed (and the constituent reactions are deleted, according to the algorithm), the section is used *intact* and is never rebuilt. This results in drastic gains in efficiency.

Even more dramatic is the case where there are extraneous reactions that lead to specialized products unrelated to the product we are pursuing. Suppose, for example, that there is in the database a series of reactions leading from a common intermediary metabolite, A, to a building block, H:

$$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G \rightarrow H$$

Since H participates in only one reaction it will be one of the first metabolites to be processed. Since there are no reactions consuming it, no combination pathways can be created; the reaction producing it is deleted, as the algorithm specifies. This modifies the chain to:

$$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G$$

The same reasoning now applies to G, which will be processed next, leading to the chain:

$$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$$

It is clear that the whole specialized irrelevant pathway will be totally eliminated early on, after it is considered by our algorithm **only once**.

The Seressiotis and Bailey algorithm[2], on the other hand, will not look at this pathway early on, because it starts from the designated substrates. By the time it reaches A (which is an intermediary metabolite and not an initial substrate), it may have already generated hundreds of pathways producing A. *For each one of these pathways*, the Seressiotis and Bailey algorithm will examine the sequence of reactions $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G \rightarrow H$ from scratch, extending each of the pathways (which reached A) all the way to H before discarding the pathways. This redundant effort will occur very often, because in enzyme databases chains of reactions that lead to specialized products are very common. The algorithm presented here, by its fundamental design an without using additional special rules, treats these chains intelligently and stands a much better chance of handling large enzyme databases efficiently.