

# Representations of Metabolic Knowledge: Pathways

Peter D. Karp and Suzanne M. Paley

Artificial Intelligence Center, SRI International

333 Ravenswood Ave., EJ229

Menlo Park, CA 94025

{pkarp,paley}@ai.sri.com

## Abstract

The automatic generation of drawings of metabolic pathways is a challenging problem that depends intimately on exactly what information has been recorded for each pathway, and on how that information is encoded. The chief contributions of the paper are a minimized representation for biochemical pathways called the *predecessor list*, and inference procedures for converting the predecessor list into a *pathway-graph* representation that can serve as input to a pathway-drawing algorithm. The predecessor list has several advantages over the pathway graph, including its compactness and its lack of redundancy. The conversion between the two representations can be formulated as both a constraint-satisfaction problem and a logical inference problem, whose goal is to assign directions to reactions, and to determine which are the main chemical compounds in the reaction. We describe a set of production rules that solves this inference problem. We also present heuristics for inferring whether the exterior compounds that are substrates of reactions at the periphery of a pathway are side or main compounds. These techniques were evaluated on 18 metabolic pathways from the EcoCyc knowledge base.

## Introduction

The EcoCyc system consists of a knowledge base that describes the genes and intermediary metabolism of *E. coli*, and a graphical user interface for accessing that knowledge. Taken together, the knowledge base (KB) and the graphical user interface (GUI) constitute an electronic encyclopedia that allows scientists to visualize an integrated collection of genomic and biochemical information.

The EcoCyc KB is stored using a frame knowledge representation system. It contains information about genes, the enzymes those genes encode, the reactions catalyzed by those enzymes, and the chemical compounds that participate in those reactions (Karp 1992; Karp & Riley 1993). In addition, the KB describes

a number of metabolic pathways, each consisting of a collection of bioreactions. The EcoCyc GUI is capable of generating displays of each of these types of objects, such as a reaction, an enzyme, or a metabolic pathway. The automatic generation of drawings of metabolic pathways is a challenging problem that depends intimately on exactly what information is recorded for each pathway, and on how that information is encoded. Our representation of metabolic pathways — such as the TCA cycle, glycolysis, and tryptophan biosynthesis — facilitates both knowledge acquisition of pathways, and automatic pathway drawing.

The chief contributions of this paper are a minimized representation for biochemical pathways called the *predecessor list*, and inference procedures for converting the predecessor list into a *pathway-graph* representation that can serve as input to a pathway-drawing algorithm. The predecessor list has several advantages over the pathway graph, including its compactness and its lack of redundancy. The conversion between the two representations can be formulated as both a constraint-satisfaction problem and a logical inference problem, whose goal is to assign directions to reactions, and to determine which are the main chemical compounds in the reaction. We present techniques for solving this inference problem. The representation and inference techniques described in this paper were evaluated for the 18 biochemical pathways in the EcoCyc KB.

The usefulness of the predecessor list for knowledge acquisition should make this representation a staple of any metabolic database project. And because the pathway graph is such a natural representation to use when reasoning about metabolic pathways, we expect the applicability of our inference procedure to extend to other computations besides pathway drawing (Karp & Mavrovouniotis 1994).

## The Pathway Representation Problem

Figure 1 is a drawing of the biosynthetic pathway for threonine biosynthesis that was generated by EcoCyc.

This drawing closely mimics those found in biochemistry textbooks.<sup>1</sup> The drawing shows a sequence of bioreactions that transform a number of reactant compounds into a number of product compounds.<sup>2</sup> Every transformation in the sequence is catalyzed by one or more enzymes, whose name(s) is drawn next to the reaction arrow (rarely, bioreactions occur spontaneously, in which case no enzyme name is drawn). The nature of a pathway is that adjacent bioreactions operate on shared compounds: the *main substrates* that lie along the backbone of the pathway (such as homoserine). The unshared compounds are the *side substrates*; they are drawn off to the side of each bioreaction, with curved arrows showing whether particular side substrates are reactants or products of a reaction.

We can represent a metabolic pathway as a graph whose nodes consist of enzymes, main substrates, and side substrates (labeled as such); the edges of the graph are the straight lines that connect main substrates, and the curved lines that connect groups of side substrates. This representation is the pathway graph. One way to formulate the pathway-drawing problem is: Given a pathway graph whose nodes and edges are typed and labeled with enzyme, main, and side information, compute display positions for each node and edge that will produce displays analogous to Figure 1. This formulation considers pathway display to be a graph-layout problem.

That formulation is the second half of the complete pathway-drawing problem. The first half of the problem is to compute the graph representation of a pathway from a minimized representation that we call the *predecessor-list* representation (see Figure 2). The following section explains why we chose to use the predecessor list as the representation stored in the EcoCyc KB. A predecessor list defines a pathway as a partial order of its component reactions. To move from the predecessor list to the graph representation, we must assign a direction to each reaction, and we must distinguish main from side substrates; the remainder of the paper describes these inferences in detail.

This paper is concerned with the first step in Figure 2; the pathway drawing algorithms of the second step are described in (Karp & Paley 1995).

<sup>1</sup>Our initial goal is to provide users with the type of drawings they are familiar with from textbooks; later we will produce more powerful displays that cannot practically be used in books.

<sup>2</sup>In this paper we use *reactants* to mean all compounds on the left side of the reaction as written; *products* are the compounds on the right side. *Substrates* are the union of products and reactants.

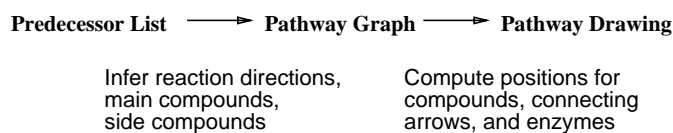


Figure 2: We compute a pathway drawing from the predecessor list using our intermediate representation called the pathway graph.

## Representation of Metabolic Pathways Design Criteria

One key design criterion for the predecessor-list representation is compactness. A terse encoding will speed the knowledge acquisition process (pathway definitions are entered manually by our collaborator, Dr. Monica Riley).

One way to encode a pathway would be to explicitly record all information in Figure 1, including the exact location of every node. But we can make the encoding more compact if we do not require that the KB explicitly store all aspects of that drawing. First, we rely on automatic layout algorithms to compute the positions of all of the components of the pathway automatically, which frees us from entering position information by hand, and from reentering position information when a pathway definition changes. Second, we assume that an algorithm can automatically distinguish between the *main substrates* along the backbone of the pathway (such as homoserine), and the *side substrates* (such as ATP). Therefore, the representation need not capture either position information or the main/side substrate distinction.

The only remaining items of information that the encoding must therefore capture are the compounds involved in each step of the pathway, the interconnections among steps, and the enzyme(s) that catalyze each step.

Another section of the EcoCyc KB already contains information about the compounds involved in each pathway step, namely the 2,800 reaction definitions in the KB. Most of these reactions were codified by the Enzyme Nomenclature Committee (Webb 1992). That reaction information is supplemented by additional information entered by Dr. Riley.

Nonredundancy is a basic principle of database and KB design, and underlies the *normal forms* of database theory (Ullman 1982). A KB with minimal redundancy should take less time to acquire, will accumulate fewer typographical errors during acquisition, will be easier to update when errors are detected (since information must be updated in only one place), and will not accumulate errors due to a user who updates only

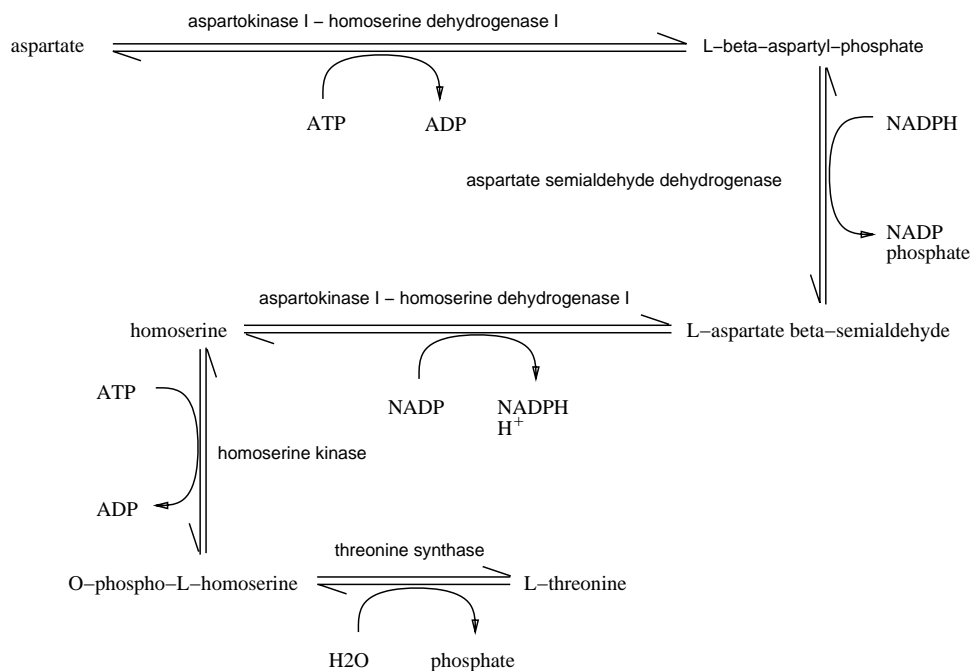


Figure 1: The *E. coli* pathway for threonine biosynthesis.

one element from a redundant set.

Therefore, we represent a pathway by reference to its component reactions. The pathway graph representation is highly redundant with respect to the reactions already encoded in the KB, because the nodes and edges (compounds and transformations) are already stored in reaction frames.

### The Predecessor List Representation

A pathway is a collection of reactions. A linear pathway such as that in Figure 1 could be represented very simply as the sequence of its component reactions, therefore it is tempting to use a reaction list as a representation for a pathway. The problem with this approach is that the same set of reactions can be molded into pathways of different topologies to emphasize different relationships. Figures 3 (a) and (b) show different pathways composed of the same set of reactions; (b) emphasizes the role of *Y* in the reaction; treating it as a main compound, whereas (a) treats *Y* as a side compound. One could argue that the techniques we later present for inferring whether exterior compounds are mains or sides could be used to decide whether pathway (a) or (b) is most appropriate for a given reaction list — by determining whether *Y* should be a main or a side. Those techniques, however, are heuristic, and since an error in inferring pathway connectivity would be considered very significant by a biologist, we have chosen to encode reaction connectivity

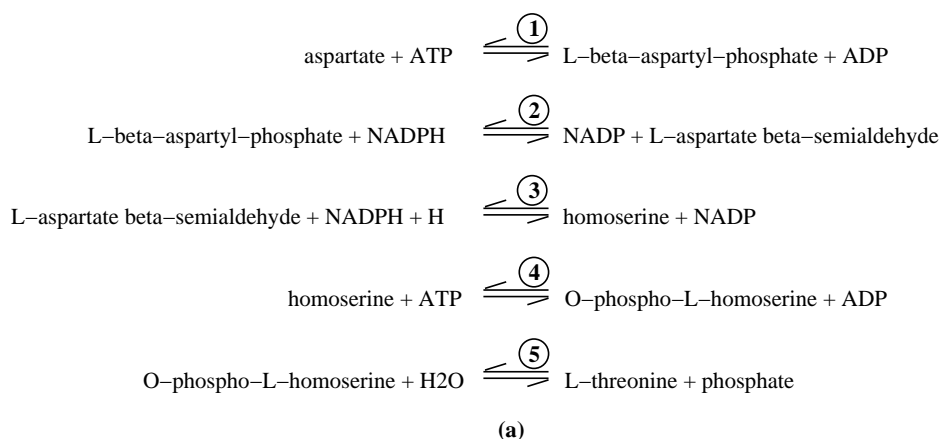
explicitly using the *predecessor-list* representation.<sup>3</sup> A pathway is encoded as a list of tuples, where each tuple is a pair consisting of a reaction, and the predecessor of that reaction in the pathway.

For example the threonine biosynthetic pathway in Figure 1 consists of the set of reactions shown in Figure 4(a), connected by the predecessor list shown in Figure 4(b). In a linear pathway, the first reaction has no predecessor, and the last reaction has no successor. In a cyclic pathway, every reaction has one predecessor. In a branched pathway, the reactions at the branch points have either more than one predecessor or more than one successor.

### Inferring Side Compounds and Reaction Direction

The predecessor list representation lacks information required for pathway drawing, so the first step in displaying a pathway is to convert the predecessor-list form of a pathway stored in the EcoCyc KB to a graph representation in which main and side compounds (henceforth referred to as *mains* and *sides*) have been identified as such, and the directed connections between the main and side compounds are known (but display positions for nodes are unassigned).

<sup>3</sup>Another reason is that as we define very large, computer-synthesized pathways, ambiguities of this sort will become more frequent.



((predecessor 2 1) (predecessor 3 2) (predecessor 4 3) (predecessor 5 4))

(b)

Figure 4: (a) The set of reactions in the biosynthetic pathway for threonine. In this figure we have drawn each reaction in the direction it occurs in the pathway, which of course will not always be the case. (b) The predecessor list for the pathway. Each number in the predecessor list refers to a reaction in (a): the first tuple encodes the fact that the predecessor of reaction 2 is reaction 1.

The main compounds lie along the backbone of the pathway—these compounds are shared between consecutive steps of a pathway.<sup>4</sup> Therefore, if reaction directions were known, main compounds would drop out immediately as the intersection of the products of reaction *A*, and the reactants of reaction *B*, when *A* is the predecessor of *B*. However, the direction recorded for a particular reaction within the EcoCyc KB may differ from the direction of that reaction in a given pathway. It is not possible to simply alter the KB definition of each reaction to use the direction required by its parent pathway because in some cases the same reaction proceeds in different directions in different pathways (the nonredundancy principle instructs us to describe each reaction only once, as opposed to creating separate entries for the two directions of the same reaction). A second reason for the constraint is that one element of the systematization compiled by the Enzyme Nomenclature Committee is a designation of reaction direction. That is, the classification system specifies the direction in which many reactions must be written. For example, all hydrolase-catalyzed reactions are written

<sup>4</sup>In fact, this definition of main compounds as shared compounds is overly restrictive. In some cases biochemists draw one or two compounds in a pathway as mains that our definition would treat as sides. They are drawn as mains to highlight them as important products or reactants in the pathway. Our approach cannot infer mains of this sort, but they can be explicitly identified.

with water as a product, not a reactant. Therefore, if we wish to maintain the EC classification system in EcoCyc, we cannot alter the stored reaction directions. We could of course manually encode the direction of a reaction in each pathway in which it occurs, but if we infer this information, we save human time.

The problem of computing the graph representation for a pathway can be stated as follows: *Given*: A predecessor-list representation of a pathway, plus the reaction equations for each reaction in the predecessor list, *Find*: a consistent assignment of directions to each reaction, a classification for every compound of either main or side in each reaction, and the enzymes that catalyze each reaction in the pathway. From this information it is trivial to compute the actual nodes in the pathway graph.

Although each reaction in the problem input already has a stored direction, that direction is ignored since, as discussed earlier, it may be incorrect for a given pathway.

The consistency criterion interrelates the assignments of directionality and of side/main status, and is the key to solving the problem. The criterion is: given two consecutive reactions *A* and *B* as defined by the predecessor list, and given a putative assignment of directions to *A* and *B*, the intersection of the products of *A* with the reactants of *B* must be non-null. Those compounds in the intersection are mains; the compounds not in the intersection are sides. More

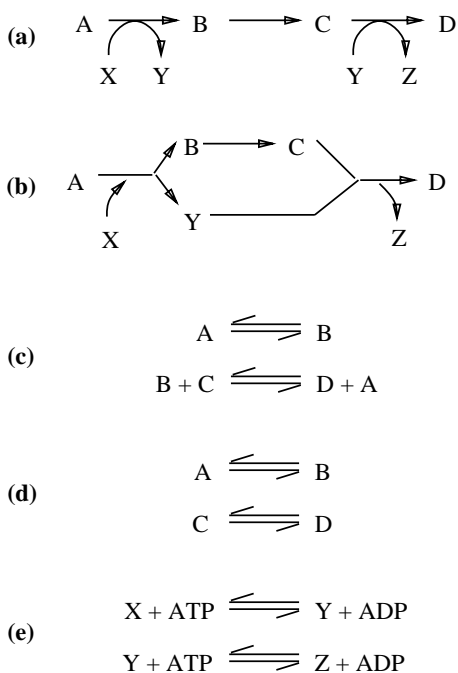


Figure 3: Example pathways. (a,b) Pathways of different connectivities that are composed of the same reactions. (c) An ambiguous pathway definition (two combinations of direction assignments give overlapping products and reactants). (d) A pathway for which no combination of direction assignments gives overlapping products and reactants. In (c) and (d) we assume the top reaction is declared to be the predecessor of the bottom reaction. (e) illustrates that cofactors can complicate the inference of reaction direction.

formally the consistency criterion is:

$$\forall A \forall B (\text{reaction } A) \wedge (\text{reaction } B) \wedge (\text{predecessor } B \ A)$$

$$\supset (\text{intersection (products } A) \text{ (reactants } B)) \neq \epsilon$$

where the products and reactants are computed in the context of the putative direction assignments for  $A$  and  $B$ .

It is possible to define pathways for which more than one assignment of directions satisfies the consistency criterion, or for which no solution exists. We consider such pathways to be incorrectly specified. Figure 3(c) and (d) show examples of reactions and predecessor lists with multiple solutions and no solutions, respectively. We extended the representation to allow more information to be added to the problem input to remove ambiguity: the user can specify for a pathway that a given compound is a main reactant or a main product of some reaction (no pathways defined in Eco-

Cyc are in fact ambiguous).

In a pathway  $P$  of  $N$  reactions, each reaction might proceed in two possible directions. Therefore  $2^N$  possible direction assignments exist for the reactions of  $P$ . A brute-force algorithm could simply generate all combinations of direction assignments, and test that each reaction in  $P$  satisfies the consistency criterion. Using this algorithm for a pathway of 12 steps such as purine biosynthesis would require testing 4096 combinations, which would not be feasible to compute dynamically during pathway display, so we developed a more efficient approach.

The algorithm we developed is based on a set of production rules that infer directions for the reactions in a pathway (see Figure 5). The top-level goal of the inference task is to derive that the direction of each reaction is known, as either left-to-right or right-to-left (left-to-right means the direction stored for the reaction in the EcoCyc KB is correct; right-to-left means the reverse of the stored direction is correct). The rules shown deduce the left-to-right case; an analogous set of rules handles the right-to-left case. All of the rules derive the direction of a given reaction  $B$  by considering the overlap between the products and reactants of  $B$  with the products and reactants of the neighboring reaction.

Rules 1 and 2 handle the case where the direction of a reaction  $A$  can be immediately determined with respect to its predecessor  $B$  because there is a non-null intersection between *either* the left or the right substrates of  $A$  (but not both), with some of the substrates of  $B$ .

For example, consider the first two reactions ((1) and (2) in Figure 4) of threonine biosynthesis (Figure 1). Rule 1 would conclude that reaction (2) should proceed in the left-to-right direction based on its overlap with its predecessor reaction (1).

Rules 1 and 2 are not sufficient because in some cases the direction of a given reaction is not determined solely by its predecessor — if, for example, there is overlap between products and reactants on both sides of both reactions. Rules 3 and 4 backward chain to determine the direction of a neighbor of a reaction  $B$ ; once the direction of a neighbor of  $B$  is known, the direction of  $B$  can often be inferred. Note that in Rule 3,  $B$  is the predecessor of  $A$ , whereas in Rule 4 the converse is true. In our experience, most reaction directions can be deduced using either Rule 1 or 2, thus the average complexity of these rules for realistic metabolic pathways in the EcoCyc KB is  $O(N)$ , in contrast to the worst-case complexity of  $O(2^N)$ .

Our implementation of this algorithm actually uses hard-coded COMMON LISP procedures rather than a

0. (known-direction ?r)  $\leftarrow$  (left-to-right ?r)  
 (known-direction ?r)  $\leftarrow$  (right-to-left ?r)
1. (left-to-right ?A)  $\leftarrow$  ((predecessor ?A ?B)  $\wedge$   
 (intersectionp (right ?B) (left ?A))  $\wedge$   
 $\neg$  (intersectionp (right ?B) (right ?A))  $\wedge$   
 $\neg$  (intersectionp (left ?B) (right ?A)))
2. (left-to-right ?A)  $\leftarrow$  ((predecessor ?A ?B)  $\wedge$   
 (intersectionp (left ?B) (left ?A))  $\wedge$   
 $\neg$  (intersectionp (right ?B) (right ?A))  $\wedge$   
 $\neg$  (intersectionp (left ?B) (right ?A)))
3. (left-to-right ?A)  $\leftarrow$  ((predecessor ?A ?B)  $\wedge$   
 (known-direction ?B)  $\wedge$   
 (intersectionp (products ?B) (left ?A))  $\wedge$   
 $\neg$  (intersectionp (products ?B) (right ?A)))
4. (left-to-right ?A)  $\leftarrow$  ((predecessor ?B ?A)  $\wedge$   
 (known-direction ?B)  $\wedge$   
 (intersectionp (reactants ?B) (right ?A))  $\wedge$   
 $\neg$  (intersectionp (reactants ?B) (left ?A)))

Figure 5: Rules for inferring reaction directions. The functions `right` and `left` return the compounds on the left and right sides of a reaction *as it is stored in the KB*. Functions `reactants` and `products` return the compounds on the left and right sides of a reaction *given a putative direction assignment by the algorithm*. The `predecessor` predicate refers directly to the predecessor list. The `intersection` predicate is true just in case the intersection of its two arguments is non-null. Rules that infer the `right-to-left` predicate are not shown, but are analogous to Rules 1–4.

production-rule interpreter. A further refinement of our algorithm is to remove compounds such as those listed in columns (a) and (b) of Table 1 from consideration as overlapping compounds between a pair of reactions, unless this step would remove all overlap between two reactions. This step eliminates multiple solutions from a number of pathways.

For example, imagine that the algorithm began by attempting to infer the direction of reaction (3) by comparison to its predecessor, reaction (2). Rules 1 and 2 would fail (as would their right-to-left counterparts) because (for Rule 2), the left side of reaction (3) overlaps with both sides of reaction (2). Rule 3 would also fail, because even if we had inferred the direction of reaction (2) to be left-to-right (as in the previous example), both sides of reaction (3) overlap with the right side of reaction (2).

The proof can go through in two ways. Our heuristic of removing cofactor compounds such as NADP and NADPH when other overlap exists (as it does here), would allow Rule 3 to succeed. Or, Rule 4 could succeed by backchaining to establish the directions of reaction (4).

This algorithm computed correct reaction directions, plus main and side compounds, for all 18 pathways in the EcoCyc KB.

## Deducing Main and Side Compounds for Initial and Final Reactions

Once reaction directions are known, the mains and sides for internal reactions in a pathway can be determined automatically from the overlap between the reactants and products of a reaction and its predecessor, respectively. However, if a reaction has no predecessors, there is no a priori means of determining which of its reactants should be mains and which should be sides. The same is true for the products of a reaction that has no successors. The products and reactants of these *external* reactions are called the *exterior* compounds; compounds that are not exterior are called interior (for example, in Figure 1 aspartate and ATP in the first reaction, plus L-threonine and phosphate in the last reaction, are the only exterior compounds). Although mains and sides cannot be rigorously determined in these cases (and there may be cases in which more than one option is acceptable), we have developed a set of heuristics that allow us to produce the correct determination most of the time.

Two simple heuristics enable us to make the correct main and side determination in 85% of the cases we have examined so far. First, there is a set of simple inorganic molecules (such as those listed in column (a) of Table 1) that will almost never be mains if other

(a)		(b)	(c)
HCO <sub>3</sub> <sup>-</sup>	H <sub>2</sub> O	ATP, ADP, AMP	glutamine, glutamate
CO <sub>2</sub>	H <sup>+</sup>	FAD, FADH <sub>2</sub>	glutamate, 2-oxoglutarate
OH <sup>-</sup>	H <sub>2</sub>	NAD, NADH	
P <sub>i</sub>	NH <sub>3</sub>	NADP, NADPH	
PP <sub>i</sub>	SO <sub>4</sub> <sup>-</sup>		

Table 1: Examples of individual compounds (a), and groups of compounds (b and c), used to infer side and main-compound status. This list is partial, omitting coenzyme-A derivatives, for example.

alternatives are available. Thus, the first heuristic is to simply remove these molecules from consideration as mains for a given side of a reaction if other alternative compounds exist.

It is tempting to place other common cofactors, such as ATP, on this list also, since such compounds usually serve as sides. However, there do exist pathways in which these compounds are mains (e.g., the synthesis of AMP from adenylo-succinate should show AMP as a main, even though it is a side in most reactions in which it participates). Our solution to this problem forms the second heuristic. We identify groups of compounds such that when one member of the group is an interior compound that has already been inferred as a side, we infer that the other member of the group is a side if it is an exterior compound of the same reaction. For example, glutamine and glutamate commonly serve as a nitrogen donor/acceptor pair. When glutamate is a known interior side product, and glutamine is one of several exteriors, we assume that glutamine is acting as a nitrogen donor and should be a side. If glutamate were not present as a side product, we would assume that glutamine has a more central role, and should be a main. Similarly, when ATP is paired with ADP or AMP, we infer that it is serving as a phosphate donor (and releasing energy) and is a side, whereas without them ATP must be contributing structurally to the product, so it is a main. Columns (b) and (c) of Table 1 list examples of other groups of cofactor compounds.

More generally, the reasoning here is that donors and acceptors of phosphate, nitrogen, and electrons should be treated as sides. Strictly speaking, implementing this reasoning as a search for groups of compounds could be flawed—for example, the presence of ATP and ADP on opposite sides of a reaction does not necessarily imply that their role in the reaction is as phosphate donor or acceptor. Strict determination of the role of a compound in a reaction requires reasoning about the exchange of chemical groups and about reaction mechanisms that is beyond our current capabilities.

So far, EcoCyc contains 18 simple pathways (plus

some larger pathways made up of several simple pathways). These pathways yield 33 reactions that lack either predecessors or successors, and for which we tested our heuristics to classify their exterior compounds as mains or sides. The preceding two heuristics gave correct answers for 28 of these reactions. Although this leaves us with the very small sample size of 5 reactions with which to evaluate additional heuristics, we believe that those we have chosen will be widely enough applicable to be useful.

There is a difference between an acceptable wrong answer and an unacceptable one. It is acceptable to mistakenly list a compound that should be a side as an additional main. It is unacceptable to have all sides and no mains. This last situation was the result our heuristics produced in one case, the synthesis of carbamoyl-phosphate from glutamine (as a nitrogen donor), ATP (as a phosphate donor), CO<sub>2</sub> and water. Even though this was a problem in only one reaction, it was necessary to decide on a plan of action to fix it. Our choice was to select as a main the compound “most similar to” the main on the other side of the reaction. To keep things simple, we use the number of carbons as our measure of similarity. In the above case, carbamoyl-phosphate has a single carbon, so CO<sub>2</sub> was selected as the main. This heuristic is applied only in the case when there would otherwise be no main.

Our final heuristic gave the correct result in 2 of the remaining 4 problem reactions. We surmise that it is unusual for a compound to serve as both a main and a side in different reactions in the same pathway. Thus, if we are left with more than one main compound after applying the previous heuristics, we check to see if any of them occur as sides elsewhere in the pathway. If so, we assume that the compound is a side in this reaction also.

These heuristics generate correct results for 31 of the 33 applicable reactions. The two remaining reactions each have two mains where they should only have one. If we knew there should only be one main, we could devise new heuristics to choose between them, but since some reactions do have more than one main, it is diffi-

cult to tell that this is the case. Our approach is to add additional assertions to the KB that explicitly identify the mains. Another approach is to develop better rules for evaluating the overall metabolic roles of different compounds—for example, in terms of how quickly they are consumed or produced by all metabolic pathways, or of how “important” they are in some metabolic context.

## Determining the Enzymes within a Pathway

The pathway graph must identify the enzyme(s) that catalyze each step in the pathway. The EcoCyc KB already allows us to determine all of the enzymes (if any) that catalyze a particular reaction in the KB. However, the inference that all of the enzymes that could in principle catalyze a reaction, do in fact catalyze that reaction in the context of a particular pathway, is sometimes flawed. The TCA cycle provides an example. The reaction that converts succinate to fumarate (Figure 6) can in principle be catalyzed by two distinct enzymes in *E. coli*: succinate dehydrogenase and fumarate reductase. But physiologically, the TCA cycle occurs only under aerobic conditions, with succinate dehydrogenase catalyzing this reaction, because the expression of fumarate reductase is inhibited under aerobic conditions. Fumarate reductase is expressed only under anaerobic conditions, when it catalyzes the reverse reaction in the pathway for anaerobic respiration. The two enzymes have different names that correspond to the two different directions of this reaction. Therefore, our pathway representation allows us to specify enzyme use information for a pathway: we can explicitly state that succinate dehydrogenase is the only enzyme that, in the TCA cycle, catalyzes the conversion of succinate to fumarate.

This example raises two questions: Should reaction direction be represented explicitly in the KB? We argued no earlier in the paper because of redundancy with stored reactions. Second, what fundamentally *is* a pathway? This example suggests that the very notion of the TCA cycle includes the cellular conditions under which it operates. If we represented those conditions explicitly, we could conceivably infer which enzymes would be active. We do not currently represent conditions explicitly because although EcoCyc does include the information about enzyme activators and inhibitors that would enable these inferences, it does not contain information on gene regulation that would let us infer if the gene encoding an enzyme is actually expressed under given conditions.

Work is under way to encode genetic regulatory information in EcoCyc. For example, we plan to encode

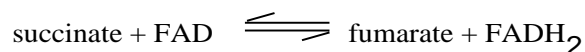


Figure 6: The reaction catalyzed by the enzymes succinate dehydrogenase (forward direction) and fumarate reductase (reverse direction).

information about the operon organization of different *E. coli* genes, the regulatory proteins that control those operons, and the effector ligands to which those proteins are sensitive. We could then run qualitative simulations that let us infer what operons are active under given cellular conditions, and therefore what enzymes are expressed and what metabolic pathways are active.

## Related Work

Reddy et al. advocate a petri-net representation for metabolic pathways for use in discrete-event simulation (Reddy, Mavrovouniotis, & Liebman 1993). This representation is very similar to the pathway graph, and could be derived from the pathway graph by converting each graph node to a petri-net place, and by deriving petri-net arc weighting from the stoichiometric coefficients stored for each EcoCyc reaction. Its similarity to the pathway graph means the petri-net has the same advantages and disadvantages with respect to the predecessor list as does the pathway graph, reinforcing the notion of the predecessor list as the preferable base representation.

We have not encountered other published descriptions of pathway representations.

## Summary of the Predecessor List Representation

Our representation of metabolic pathways consists of the predecessor list plus optional annotations that can supplement predecessor lists that are ambiguous to our reasoner. For a given pathway those annotations can state the reaction directions for one or more reactions, they can state which of several exterior compounds are main compounds, and they can state which of several potential enzymes actually catalyze a particular reaction in the pathway.

## Conclusions

Minimality is an important principle for the design of information systems: it decreases knowledge acquisition time and can decrease the complexity of maintaining a KB. The cost of minimality is that a minimized representation may not have all the information required by a consuming application. A reasoner can

flesh out a minimized representation to produce a full-bodied description that has greater utility.

We described a minimized representation for biochemical pathways called the predecessor list. It does not contain the redundancy that exists between the pathway-graph representation and the reactions stored in the EcoCyc KB. However, the pathway graph is useful as an input to pathway drawing algorithms; therefore we presented an algorithm that computes the pathway graph representation from the predecessor list. The algorithm uses production rules to infer the directions of reactions in the pathway based on the overlap of product and reactant compounds in adjacent reactions. We also presented heuristics for inferring whether the exterior compounds that are substrates of reactions at the periphery of a pathway are sides or mains. These heuristics check for pairings of specific compounds (such as NAD and NADH) in a given reaction. The predecessor list and the associated algorithms were verified on 18 pathways from the EcoCyc knowledge base.

## Acknowledgments

We are extremely grateful to Monica Riley for many stimulating discussions on these topics during our collaboration on EcoCyc. Evgeni Shevelev also contributed helpful comments. This work was supported by grant 1-R01-RR07861-01 from the National Center for Research Resources. The contents of this article are solely the responsibility of the authors and do not necessarily represent the official views of the National Institutes of Health.

## References

- Hunter, L.; Searls, D.; and Shavlik, J., eds. 1993. *Proc. of the First International Conference on Intelligent Systems for Molecular Biology*. Menlo Park, CA: AAAI Press.
- Karp, P., and Mavrovouniotis, M. 1994. Representing, analyzing, and synthesizing biochemical pathways. *IEEE Expert* 9(2):11–21.
- Karp, P., and Paley, S. 1995. Automated drawing of metabolic pathways. In Lim, H.; Cantor, C.; and Robbins, R., eds., *Proc. of the Third International Conference on Bioinformatics and Genome Research*, 225–238. World Scientific Publishing Co. See also URL <ftp://ftp.ai.sri.com/pub/papers/karp-bigr94.ps.Z>.
- Karp, P., and Riley, M. 1993. Representations of metabolic knowledge. In Hunter et al. (1993), 207–215.
- Karp, P. 1992. A knowledge base of the chemical compounds of intermediary metabolism. *Computer Applications in the Biosciences* 8(4):347–357.
- Reddy, V.; Mavrovouniotis, M.; and Liebman, M. 1993. Petri net representations in metabolic pathways. In Hunter et al. (1993), 328–336.
- Ullman, J. 1982. *Principles of Database Systems*. Potomac, MD: Computer Science Press, 2 edition.
- Webb, E. C. 1992. *Enzyme Nomenclature, 1992: Recommendations of the nomenclature committee of the International Union of Biochemistry and Molecular Biology on the nomenclature and classification of enzymes*. Academic Press.