Bulletin of Mathematical Biology (1998) **60**, 815–856
Article No. bu970043

Designing Metabolism: Alternative Connectivities for the Pentose Phosphate Pathway

JAY E. MITTENTHAL
Center for Complex Systems Research,
Beckman Institute,
University of Illinois at Urbana-Champaign,
Illinois, U.S.A.
Department of Cell and Structural Biology,
University of Illinois at Urbana-Champaign,
Illinois, U.S.A.

AO YUAN, BERTRAND CLARKE
Department of Statistics,
University of British Columbia,
Canada

ALEXANDER SCHEELINE
Center for Complex Systems Research,
Beckman Institute,
University of Illinois at Urbana-Champaign,
Illinois, U.S.A.
Department of Chemistry,
University of Illinois at Urbana-Champaign,
Illinois, U.S.A.

We present a method for generating alternative biochemical pathways between specified compounds. We systematically generated diverse alternatives to the nonoxidative stage of the pentose phosphate pathway, by first finding pathways between 5-carbon and 6-carbon skeletons. Each solution of the equations for the stoichiometric coefficients of skeleton-changing reactions defines a set of networks. Within each set we selected networks with modules; a module is a coupled set of reactions that occurs more than one in a network. The networks can be classified into at least 53 families in at least seven superfamilies, according to the number, the input–output relations, and the internal structure of their modules. We then assigned classes of enzymes to mediate transformations of carbon skeletons and modifications of functional groups. The ensemble of candidate networks was too large to allow complete determination of the optimal network. However, among the networks we studied the real pathway is especially favorable in several respects. It has few steps, uses no reducing or oxidizing compounds, requires only one ATP in one direction of flux, and does not depend on recurrent inputs.

0092-8240/98/050815 + 42 $30.00/0

1. INTRODUCTION

Although the pattern of reactions in many metabolic networks has been known for decades, usually it is unclear what alternative networks might perform the same overall transformation of metabolites. It is desirable to characterize these alternatives, in order to understand how natural selection may have arrived at particular networks and maintained them. An ensemble of alternatives also provides a basis for designing modifications of metabolism and assessing their relative merits. Here, we offer a new and general procedure to generate a large class of alternative networks that perform a given transformation. We demonstrate this procedure by generating alternatives to the pentose phosphate pathway. (A pathway is a network between an initial and a terminal metabolite.) Our aim was to see what families of alternatives would emerge, and to understand whether the real pentose phosphate pathway is optimal in some sense, or is one of several good alternatives.

In designing a network the first problem is to specify the task it must perform and the resources available. A network consists of a set of linked reaction steps, with outputs of some steps used as inputs for others. Here, a step is a transformation of metabolites catalyzed by an enzyme; a network will be called smaller if it has fewer steps. A favorable network makes relatively low demands on currency metabolites of a cell, its resources of energy and reducing power such as ATP (adenosine 5′-triphosphate) and NADH (nicotinamide adenine dinucleotide, reduced form).

Several constraints favor networks with fewer kinds of enzymes and fewer steps. Networks that use fewer kinds of enzymes require less of the limited coding capacity of the genome (Kirkwood *et al*., 1986). The concentration of proteins is limited by their solubility, if they are in solution (Brown, 1991), or by the limited number of binding sites for them on filaments or membranes. Limits on the concentration of proteins favor networks that use fewer steps to accomplish the same activity, because a given number of enzyme molecules can run a pathway more times per second if fewer of them are required per run. Similarly, the limited solubility and osmolarity of metabolites (Atkinson, 1969) favor using fewer intermediate metabolites with higher concentrations to speed flux through a network. Such networks will often have fewer sequential steps and so will tend to generate a product faster (Meléndez-Hevia *et al*., 1994).

The connectivity of a network—the set of linkages from the output of one reaction to the input of another—depends on the task the network performs and on the enzymes and metabolites allowed. Next in the design process, we specify the set of enzymes available and generate alternative connectivities compatible with the available enzymes. Here we focus on designing connectivity and evaluating indices of performance associated with it—the number and kinds of enzymes used in the network, the number of steps, the number of kinds of currency metabolites. These networks must be evaluated further with respect to the kinetics, conditions for optimal activity, and regulation of the enzymes involved.

Is the network with the fewest steps necessarily biologically optimal? That is,

could a network with more than the minimal number of steps generate a product faster than the minimal network? The answer to this question is unclear. However, Meléndez-Hevia *et al*. (1994, Theorem 1) argue that if all the enzymes in two pathways operating simultaneously have comparable kinetic parameters, then the shorter path carries more flux than the longer one. Under these conditions we conjecture that it is possible to regulate the shorter pathway and to synthesize its enzymes so that it will continue to carry more flux. So, in this case, the pathway with the fewest steps would remain optimal after its kinetic parameters and regulation have been determined.

Our method decomposes the task of generating possible networks into two stages. The first stage deals with the transformation of carbon skeletons, ignoring functional groups. In the second stage functional groups are restored in ways compatible with the task to be performed and the enzymes available to perform it. This method extends previous approaches to designing connectivity (Seressiotis and Bailey, 1988; Happel *et al*., 1990; Mavrovouniotis *et al*., 1990; Meléndez-Hevia, 1990; Mittenthal *et al*., 1993; Nuño *et al*., 1997). Most of these analyses assumed the use of naturally occurring enzymes. However, genetic engineering opens the possibility of designing enzymes that do not occur naturally, and introducing genes for these enzymes into cells. Our method accommodates this possibility by using classes of enzymes, with each class defined by the functional groups its enzymes require for activity.

In the following sections we first describe our strategy for generating alternative networks. Section 2 presents the method of generating transformations of carbon skeletons, and the results of applying this method to the pentose phosphate pathway. The method of restoring functional groups, and the resulting characterization of alternatives to the pentose phosphate pathway, is given in Section 3. The discussion in Section 4 compares this approach with other methods and explores implications of our work. The Appendices show the algorithms that we used to implement the strategy.

2. STRATEGY

The key metabolites include building blocks for constructing polymers and membranes; sources of free energy, carbon skeletons, and nitrogen; and currency molecules that transfer free energy or reducing power. The task of a network is to convert one set of key metabolites to another. The conversion among key metabolites proceeds through intermediate metabolites, by means of enzymes, through a route specified by the connectivity.

We searched for alternative networks that perform the same conversion of key metabolites, through six stages.

(1) We simplified all metabolites to the carbon skeletons, ignoring functional groups that do not contain carbon. We define a set of reactions called changers;

these are all of the possible bimolecular transformations among the allowed carbon skeletons. From the changers we generated an ensemble of C-nets—networks that accomplish the transformation between the carbon skeletons of key metabolites. We did this by solving equations that relate the stoichiometry of the overall conversion to the stoichiometry of the individual changers. The solutions are vectors of integers that specify the number of times each changer is used in a network.

(2) Each vector corresponds to one or more possible C-nets. If there are many possible C-nets for a vector we select a small set of these, usually one or two, for further consideration. The selection is based on the repeated occurrence within a C-net of a group of coupled reactions, called a module. Modules help to reduce the number of kinds of enzymes used.

(3) Addition of functional groups to a C-net gives an R-net, or realistic network. This procedure requires the introduction of connector reactions. Connectors convert the output of one changer to the input of another or to a final product, without altering carbon skeletons. An R-net is a biologically plausible network of changer and connector steps with all functional groups specified. (Note that each connector step is a single reaction. In general a sequence of connector steps is required to convert an output of one changer to an input of another.)

(4) To convert C-nets to R-nets, we first compiled a list of generic enzymes (genzymes) that could perform the changer reactions. Each genzyme is specified in terms of the pattern of functional groups in its substrates that are required for its activity. We made all possible assignments of genzymes to the changers of selected C-nets.

(5) The assignment of enzymes to changers fixes some functional groups. The identity of the key metabolites fixes other functional groups. We specified the remaining functional groups so they change minimally in the connector steps, by searching through alternative assignments of functional groups.

(6) The preceding steps were implemented by computation. By hand we compensated for limitations on the algorithms. We adjusted the type and sequence of changes in functional groups to make R-nets more realistic biologically and chemically, and we evaluated the number of connector steps.

The rationale for neglecting functional groups initially is that in many metabolic pathways the number of changer steps, which modify carbon skeletons, is appreciably smaller than the number of connector steps, which modify functional groups. That is, the ensemble of C-nets is much smaller than the ensemble of R-nets. Thus we could seek favorable R-nets through several stages of elaboration and pruning. We first generated solution vectors, and pruned the resulting C-nets to a relatively small set. We then elaborated a fraction of these C-nets, generating a collection of R-nets from each C-net. The most promising of these R-nets were adjusted by hand.

3. THE PENTOSE PHOSPHATE PATHWAY: C-NETS

We examined the pentose phosphate pathway because interconversion of 6-carbon and 5-carbon molecules requires a network with several enzymes and steps; a single changer does not suffice. The pentose phosphate pathway has been studied intensively, and the optimality of its connectivity has been examined carefully for a limited set of enzymes (Wood, 1985; Meléndez-Hevia, 1990; Meléndez-Hevia *et al*., 1994; Nuño *et al*., 1997). This previous work is the basis for our survey with more diverse enzymes.

3.1. ***Procedure for generating C-nets.*** To seek alternatives to the pentose phosphate pathway, in the first stage we deal only with carbon skeletons. We regard the key metabolites as a 6-carbon (C6) chain and a 5-carbon (C5) chain, with no functional groups. We first considered the ways to interconvert 5 C6s and 6 C5s, using a set of changers. The set consists of unidirectional bimolecular reactions that convert unbranched carbon compounds having from one to seven carbon atoms, because all the compounds in the real pentose phosphate pathway are in this set. This restriction sharply curtails the combinatorial explosion that would occur if longer carbon chains were allowed. These 68 changers (34 forward and 34 reverse reactions) are the ways to lyse or condense the compounds in such a way that the products are also compounds in the set. For example, the changer $36 \rightarrow 45$ converts a 3-carbon compound (a C3) and a C6 to a C4 and a C5, by transfer of one or two carbons. Thus the notation $ij \rightarrow km$ means the changer converts molecules having i and j carbon atoms into molecules having k and m carbon atoms. Table 1 shows the changers.

Next we solved equations relating the stoichiometry of the overall reaction, 5 C6 $=$ 6 C5, to the stoichiometry of the 68 changers. The solutions can be represented as a vector with 34 entries—a 34-vector. Each entry is the number of steps in which a changer is used in the forward or reverse direction. The number of steps is an integer, positive for forward changers, or negative for reverse changers. We only considered 34-vectors with four or fewer nonzero entries, because the real pathway uses four changers. Appendix 1 presents the algorithm we used to compute solution vectors, starting with an overall reaction and decomposing it to changers.

The possibility of futile cycles complicated the process of solution. In a futile cycle a set of molecules react together in such a way as to regenerate the set. This process may occur an arbitrary number of times, giving arbitrarily large networks. To isolate networks with futile cycles as small as possible (hopefully, absent), we imposed a subsidiary optimality criterion, forcing 34-vectors to be as close to the zero vector as possible. The procedure we used is given in Appendix 1. Happel *et al*. (1990) appear to have a different procedure for eliminating futile cycles, based on omitting reactions that contribute to futile cycles.

Table 1. The 68 changer reactions. The notation $ij \rightarrow km$ designates the changer that converts molecules having i and j carbon atoms into molecules having k and m carbon atoms. Note that each reaction labeled with a positive reaction number is followed by the reverse reaction, labeled with the negative of that number. The genzymes that can implement each changer reaction are listed, using the number labels in Table 3. The label 15* means that the reaction requires the transketolase that can transfer one or more carbons, TK_1; the transketolase that can transfer two or more carbons, TK_2, does not suffice.

Reaction number	Reaction	Genzymes
1	$11 \rightarrow 2$	4
−1	$2 \rightarrow 11$	2, 3, 9, 10
2	$12 \rightarrow 3$	4, 5, 11, 13
−2	$3 \rightarrow 12$	1, 2, 3, 6, 7, 8, 9, 10
3	$22 \rightarrow 13$	12, 14, 15
−3	$13 \rightarrow 22$	14, 15
4	$13 \rightarrow 4$	4, 5, 11
−4	$4 \rightarrow 13$	1, 2, 3, 6, 9, 10
5	$22 \rightarrow 4$	11, 13
−5	$4 \rightarrow 22$	6, 7, 8, 9, 10
6	$14 \rightarrow 23$	14, 15
−6	$23 \rightarrow 14$	12, 14, 15
7	$33 \rightarrow 24$	14, 15*
−7	$24 \rightarrow 33$	14, 15*
8	$14 \rightarrow 5$	4, 5, 11
−8	$5 \rightarrow 14$	1, 2, 3, 6, 9, 10
9	$23 \rightarrow 5$	11, 13
−9	$5 \rightarrow 23$	6, 7, 8, 9, 10
10	$24 \rightarrow 15$	12, 14, 15, 16
−10	$15 \rightarrow 24$	14, 15, 16
11	$34 \rightarrow 25$	14, 15
−11	$25 \rightarrow 34$	14, 15
12	$33 \rightarrow 15$	12, 15
−12	$15 \rightarrow 33$	15
13	$44 \rightarrow 35$	14, 15*
−13	$35 \rightarrow 44$	14, 15*
14	$15 \rightarrow 6$	4, 5, 11
−14	$6 \rightarrow 15$	1, 2, 3, 6, 9, 10
15	$24 \rightarrow 6$	11, 13
−15	$6 \rightarrow 24$	6, 7, 8, 9, 10
16	$33 \rightarrow 6$	11
−16	$6 \rightarrow 33$	6, 9, 10
17	$25 \rightarrow 16$	12, 14, 15, 16
−17	$16 \rightarrow 25$	14, 15, 16
18	$34 \rightarrow 16$	12, 15, 16
−18	$16 \rightarrow 34$	15, 16
19	$44 \rightarrow 26$	15
−19	$26 \rightarrow 44$	15
20	$35 \rightarrow 26$	14, 15, 16
−20	$26 \rightarrow 35$	14, 15, 16

Table 1. Continued

Reaction number	Reaction	Genzymes
21	45 → 36	14, 15
−21	36 → 45	14, 15
22	55 → 46	14, 15*
−22	46 → 55	14, 15*
23	16 → 7	4, 5, 11
−23	7 → 16	1, 2, 3, 6, 9, 10
24	25 → 7	11, 13
−24	7 → 25	6, 7, 8, 9, 10
25	34 → 7	11
−25	7 → 34	6, 9, 10
26	26 → 17	12, 14, 15, 16
−26	17 → 26	14, 15, 16
27	36 → 27	14, 15, 16
−27	27 → 36	14, 15, 16
28	46 → 37	14, 15, 16
−28	37 → 46	14, 15, 16
29	56 → 47	14, 15
−29	47 → 56	14, 15
30	66 → 57	14, 15*
−30	57 → 66	14, 15*
31	35 → 17	12, 15, 16
−31	17 → 35	15, 16
32	45 → 27	15, 16
−32	27 → 45	15, 16
33	55 → 37	15
−33	37 → 55	15
34	44 → 17	12, 15, 16
−34	17 → 44	15, 16

A given 34-vector usually corresponds to more than one C-net, because most 34-vectors have some entries with absolute value greater than or equal to 2. This means that changers will often occur more than once, and the products they make will be generated at several locations in the C-net. In turn, these products may enter several other changers. Consequently there would then be more than one way to couple changers, leading to diverse non-isomorphic connectivities—that is, to diverse C-nets. Some connectivities have modules, while others do not. (Recall, a module is a cluster of coupled steps that occurs two or more times in a C-net). We examined all of the alternative modularizations of each 34-vector.

The C-net for the pentose phosphate pathway, shown in Fig. 1, illustrates the concept of modules. This C-net has seven steps and two modules; the steps not in modules constitute the remainder. These are two copies of a module with three steps, and a remainder with one step. In the module, changer 36 → 45 produces a C4 that is used in changer 46 → 37; the outputs of the latter changer are used in changer 37 → 55. The remainder, changer 6 → 33, provides a C3 as input to each

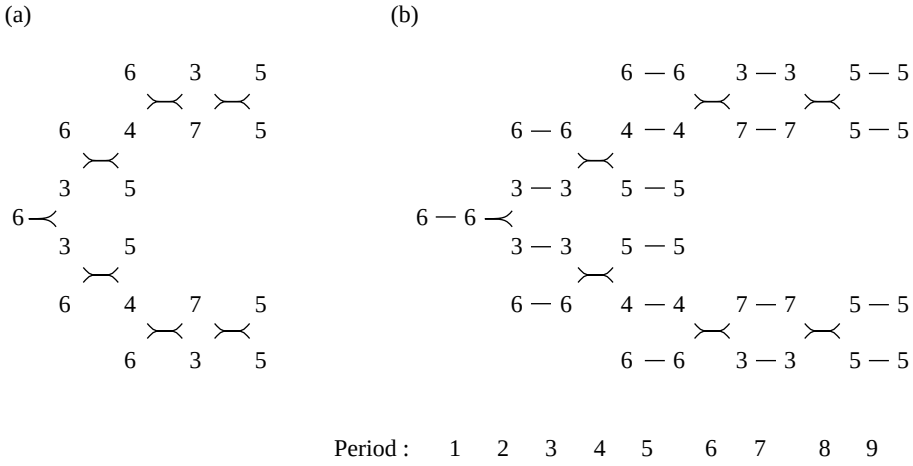


Figure 1. (a), The C-net for the real pentose phosphate pathway. It has two copies of a module having three steps, and a remainder with one step. In the module, changer 36 → 45 produces a C4 that is used in changer 46 → 37; both of these outputs are used in changer 37 → 55. The remainder, changer 6 → 33, provides input to each module. (b), The C-net in (a) with connections (sequences of connector reactions) indicated by lines. The periods are fictive time intervals during which the indicated reactions occur in parallel. Note that the periods are defined so that sequentially dependent reactions occur in successive periods.

module. In the real pathway these C3s are different compounds, glyceraldehyde 3-phosphate and dihydroxyacetone phosphate, which require different connectors to feed into the two modules.

Of the C-nets compatible with a 34-vector, we only used the C-nets with the most reactions in modules—maximally modular networks—because they are more likely to occur naturally. This is so because the same enzymes can be used for changers and connectors in every occurrence of a module within an R-net. Thus less coding capacity is required than in an R-net that uses different enzymes in different occurrences of the module. Note that selection for maximally modular C-nets does not necessarily choose a unique C-net from a 34-vector; there may be two or more such C-nets, in which changers are allocated to modules in different ways. Appendix 2 formalizes our procedure for finding maximally modular networks.

3.2. ***Characteristics of C-nets.*** The procedure in Appendix 1 produced 275 34-vectors. Of these, two use three changers, and the rest four. The method in Appendix 2 gave 606 maximally modular C-nets, which we diagrammed as in Fig. 1. This collection of C-nets can be decomposed into 53 disjoint families. Two C-nets are in the same family if and only if they have the same number of modules, the same inputs and outputs for the remainder, and the same inputs and outputs for a module. They differ in the reactions occurring within the remainder and/or the modules. For example, the real pentose phosphate pathway is one of 83 C-nets in family 1, the largest family. All C-nets in this family have a remainder with a C6

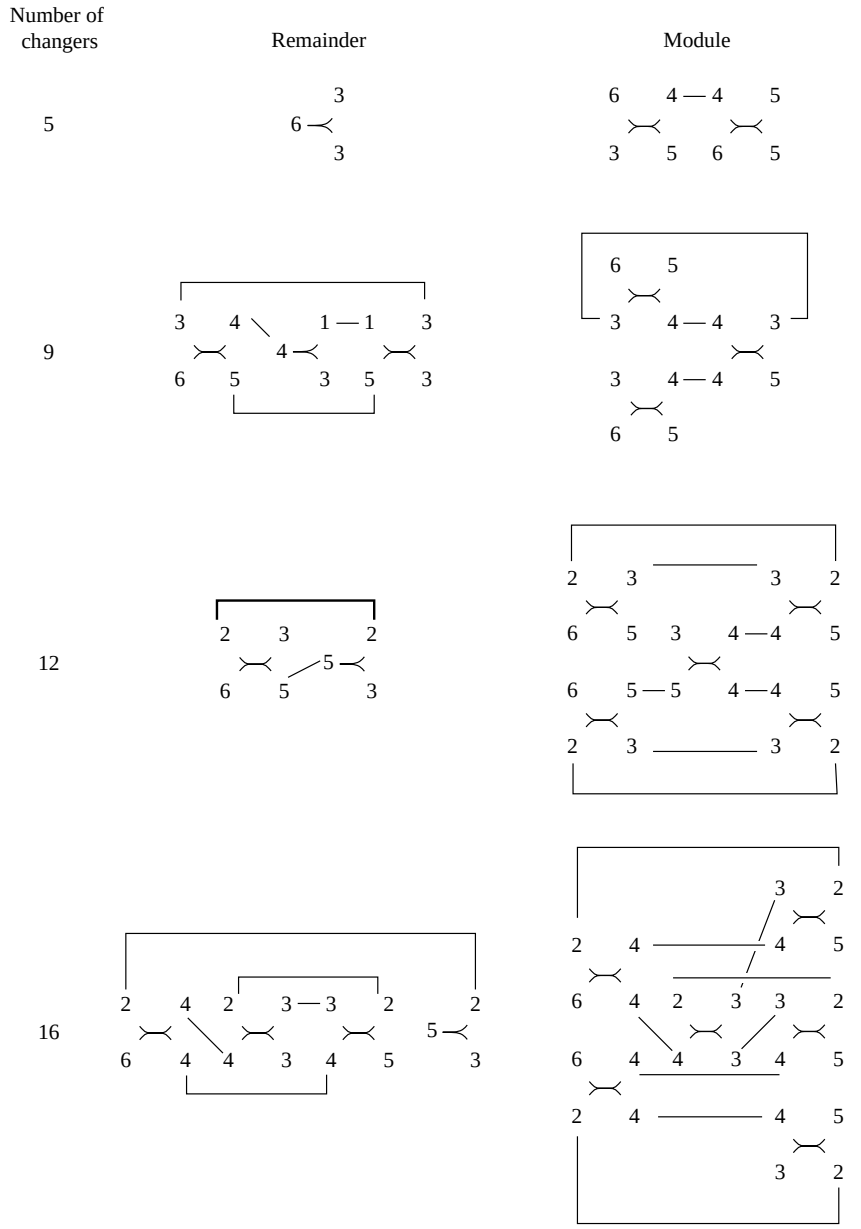


Figure 2. Diverse internal structures for remainders and modules in family 1. The C-net with five changers is the smallest C-net in family 1. Comparison with Fig. 1(a) shows that the last two changers in the module of the real C-net have been condensed to one changer in the five-step C-net. The nine-step C-net illustrates the appearance of a recurrent 3 in the remainder, and a recurrent 3 in the module. The 12-step and 16-step C-nets have several recurrents.

as input and two C3s as output ($6 \rightarrow 33$), and two identical modules, each with two C6s and a C3 as input, and three C5s as output ($663 \rightarrow 555$). However, these remainders and modules have diverse internal structures, as illustrated in Fig. 2. In family 1 we found 21 kinds of remainder and 34 kinds of module. Some conceivable combinations of remainder and module did not occur together in a C-net, because it would have had more than four kinds of changers. We found that a 34-vector could generate maximally modular C-nets in two or more families.

The 53 families of C-nets segregate into seven superfamilies. Two families are in the same superfamily if and only if they have the same number of modules, and the linkages among the modules and the remainder fit a common pattern. For example, Fig. 3(a) shows the generic connectivity of networks in superfamily 2.1, and Figs 3(b)–(d) show three of the ten families in this superfamily. Table 2 shows the common pattern for each superfamily, and lists the inputs and outputs for the remainder and the modules, and the number of C-nets, for the families in each superfamily. The caption of Table 2 also shows the equations that generate these taxa. Note that the superfamilies, families, and C-nets per family that we generated were limited by the restriction to four or fewer changers. Table 2 also shows, for each family the minimum number of steps in a C-net. Families 1, 2 and 6 were the only ones in which five changer steps sufficed to interconvert 5 C6 and 6 C5. All of these families are in the same superfamily, 2.1. In all the other families this interconversion required six or more changer steps.

A recurrent compound is a compound that is generated within a C-net but is required at an earlier stage to run the C-net. If the recurrent is not supplied, the C-net cannot run. A recurrent limits flux in either direction through a pathway. A recurrent may be consumed and produced within the remainder or within a module, or may link the remainder and a module (Figs 2 and 3 provide examples). Most C-nets that we found have one or more recurrent compounds, though some lack recurrents. In C-nets that require recurrents, the available concentrations of the recurrent compounds affect the rate at which the pathway operates, in either direction. For recurrents other than C5 and C6, the concentration of the recurrent limits the flux rate. To avoid such limitation another pathway, different from the network that has recurrents, must adjust the supply of the recurrents. (For example, oxaloacetate is a recurrent for the tricarboxylic acid cycle, but it can be supplied independently from glucose by carboxylation of pyruvate). By contrast, C6 and C5 compounds are likely to be sufficiently abundant that their concentrations do not limit flux if they are recurrent. So, we only considered C-nets with recurrent C5 or C6.

4. THE PENTOSE PHOSPHATE PATHWAY: R-NETS

We regard the pentose phosphate pathway as converting five D-fructose 6-phosphate (F6P) molecules to six D-ribose 5-phosphate (R5P) molecules, or vice versa. (This is somewhat arbitrary. Alternative C6 and C5 sugars could be chosen, with

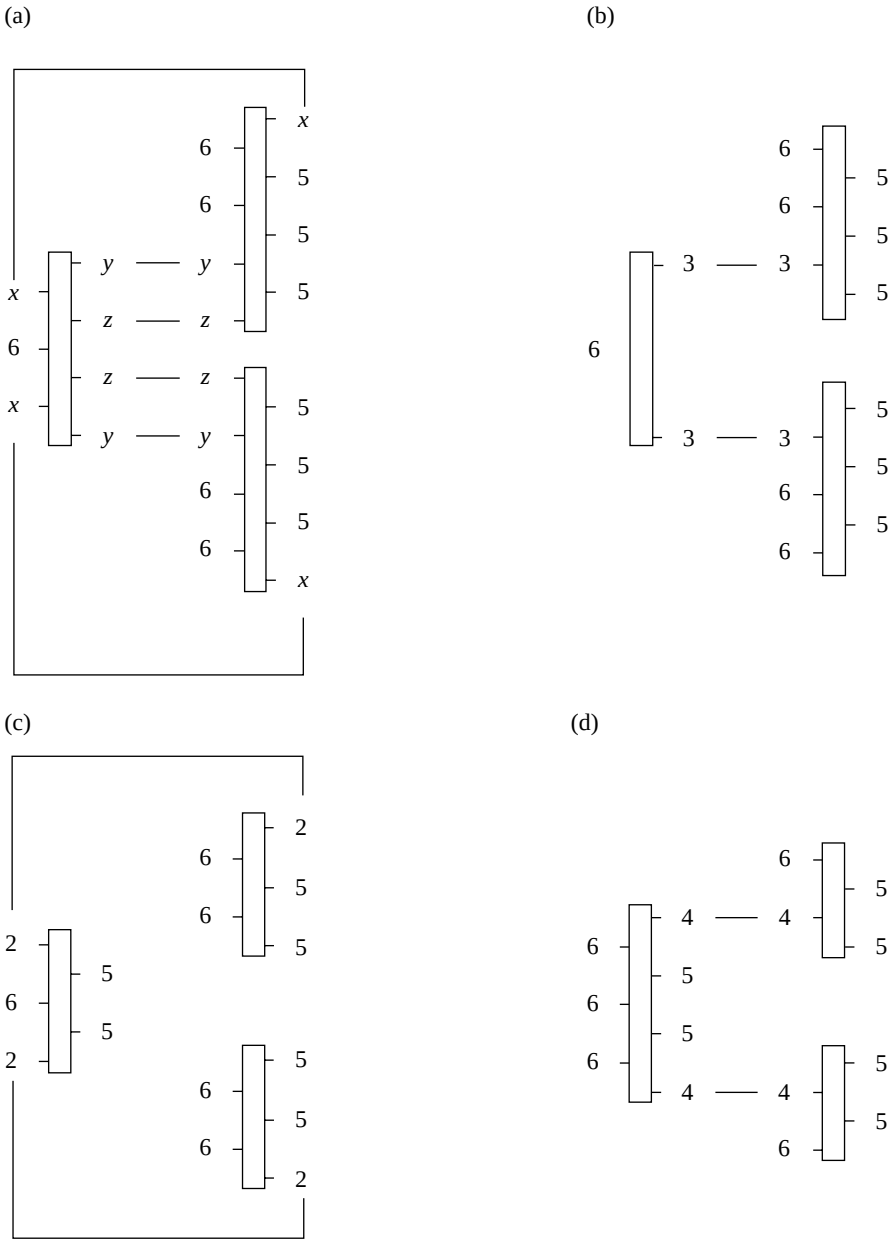


Figure 3. Generic connectivity of networks in superfamily 2.1, and 3 of the 10 families in it. The rectangles represent the internal organization of the remainder or of a module. (a), Superfamily 2.1; x is a recurrent from the module to the remainder, y and z are outputs of the remainder and inputs to the module, and x, y, and z range from 0 to 7 but are constrained by stoichiometry, as described in the caption of Table 2. (b), Family 1; it has no recurrents and includes the real pathway; $x = 0, y = 3, z = 0$. (c), In Family 9, $x = 2$ does not act as a recurrent if the modules run before the remainder; $y = 5, z = 0$. (d), Family 2 has no recurrents; because $x = 6$, the module only requires one 6 as an input, rather than the two 6s apparent in Fig. 3(a); $y = 4, z = 5$.

minor changes in the number of steps needed for conversion). We assume that the pathway can operate in either direction, depending on the relative concentrations of F6P and R5P. This seems reasonable, as all reactions in the pathway are reversible except phosphorylation of F6P to fructose 1,6-*bis*phosphate.

In this section we show the results of converting selected C-nets to R-nets. We assigned genzymes to changer reactions in all possible combinations. For each combination we generated an R-net by finding the assignment of functional groups that minimized the total number of functional group changes, compatible with the assignment of genzymes and with the use of R5P and F6P. In the R-nets with relatively few steps, we then determined the number of steps and adjusted the pattern of functional group changes to be more realistic.

4.1. ***Procedure for generating R-nets.*** Recall that an R-net is a realistic network of changers and connectors with all functional groups specified, whereas a C-net is a network of changers only. Starting with a selected C-net, we assigned genzymes to each changer in all possible ways, using the algorithm in Appendix 3. As each genzyme is defined by specifying the functional groups required for its activity, this is the first step toward converting a C-net to an R-net. Figure 4 shows an example of the specification of functional groups required for a genzyme. Table 3 shows the list of 16 genzymes that we used. Note that Table 1 lists the genzymes that can implement each of the 68 changer reactions. This list takes into account the restrictions in Table 3 on the carbon skeletons on which genzymes can act.

The assignment of genzymes and the use of F6P and R5P constrain the subsequent choice of other functional groups. Diverse choices are compatible with these constraints. However, we determined the unspecified functional groups so that the minimum number of changes in functional groups occurs. We implemented this criterion of minimization using an algorithm in Appendix 3. To simplify the determination of the unknown functional groups, initially we assumed that any single-bonded functional group can be converted to any other one. Such a conversion is called one functional group change. Double bonds to heteroatoms were counted as two single bonds to two functional groups.

Networks that compared favorably with the real one, with respect to the number of functional groups changes, were then adjusted by hand to make them more plausible biologically. The criterion of performance is the total number of steps—changer steps plus connector steps. Therefore, the first task was to determine the number of connector steps in the R-net. Some changes of a single functional group require several steps. Examples are transitions between a hydrogen and a hydroxyl group, or a hydrogen and a phosphate group. Some ways to modify a double-bonded functional group require one enzyme (as in reduction of C=0 to H–C–OH), others two enzymes.

Table 2. Inputs and outputs of remainders and modules for families, classified by superfamilies. Each superfamily has a label of the form $a.b$, where a is the number of modules that each family in the superfamily has, and b is the bth superfamily with that number of modules. In all families of a superfamily the following parameters are constant: $m =$ number of modules; $s =$ number of C6s that are inputs to each module; $n =$ number of C6s for remainder (< 0 if inputs, > 0 if outputs). Since five C6s are inputs to the network, we have $m^*s - n = 5$. Let p denote the number of output C5s from the remainder. Although p is not constant over a superfamily, a group of families within a superfamily have the same value of p. In a particular family, let w and x denote the number of carbons in two recurrent inputs to the remainder, and let y and z denote the number of carbons in outputs from the remainder to each module, other than C6s and C5s characteristic of the family. Now, conservation of carbon atoms in the remainder implies $m(w+x) = m(y+z) + 6n + 5p$. The table is arranged as follows. Each superfamily label is followed by the numerical values of m, s, n, and p, and by the conservation equation, for a group of families with the same p within the superfamily. Within a superfamily the families are listed by dictionary order on y and z. For each family the number of carbon atoms in the inputs (in) and outputs (out) of the remainder and the module are given. These are followed by the number of C-nets in the family and the minimum number of steps in any C-net of the family. The last column indicates whether recurrent compounds between the remainder and the module are present (y) or not (n) in each family. (In some families without recurrents the modules must operate before the remainder.) For superfamily 2.1, Fig. 3 illustrates the meaning of the numbers of carbons x, y, and z, and displays graphically the rows of this table for families 1, 2, and 9.

	Remainder		Module		Number of	Minimum	Recurrent
Family	In	Out	In	Out	C-nets	steps per C-net	compounds?
Superfamily 2.1: $m = 2, s = 2, n = -1, p = 0; 2x = 2(y+z) - 6$							
	$x6x$	$yzzy$	$66yz$	$x555$			
1	060	3003	6630	0555	83	5	n
13	161	4004	6640	1555	18	6	y
14	262	4114	6641	2555	8	6	y
15	363	4224	6642	3555	7	6	y
32	565	4444	6644	5555	5	7	y
9	262	5005	6650	2555	44	6	n
11	363	5115	6651	3555	16	6	y
16	464	5225	6652	4555	17	6	y
2	666	5445	6654	6555	27	5	n
6	767	5555	6655	7555	7	5	n
Superfamily 2.2: $m = 2, s = 3, n = 1, p = 0; 2(w+x) = 2(y+z) + 6$							
	$wxxw$	$yz6zy$	$666yz$	$wx555$			
28	3003	00600	66600	30555	19	7	y
38	1221	00600	66600	12555	1	11	y
40	2222	10601	66610	22555	1	11	y
47	3443	22622	66622	34555	1	11	y
48	4444	23632	66623	44555	1	11	y
33	5005	20602	66620	50555	1	8	y
18	7007	40604	66640	70555	9	6	y

Table 2. Continued

Superfamily 3.1: $m = 3, s = 1, n = -2, p = 0; 3(w + x) = 3(y + z) - 12$							
	$6xxx6$	$yyyzzz$	$6yz$	$x55$			
52	62226	333333	633	255	1	13	y
3	60006	444000	640	055	74	6	n
19	61116	555000	650	155	19	7	n
22	62226	555111	651	255	7	8	y
25	63336	555222	652	355	17	7	y
29	64446	555333	653	455	11	7	y
For $p = 6$:							
	$6wwwxxx6$	$555555yyy$	$6y$	wx			
34	63337776	555555444	64	37	1	8	y
Superfamily 3.2: $m = 3, s = 2, n = 1, p = 0; 3(w + x) = 3(y + z) + 6$							
	$wwwxxx$	$6yyyzzz$	$66yz$	$wx55$			
39	111111	6000000	6600	1155	1	11	y
10	222000	6000000	6600	2055	8	8	y
12	333000	6111000	6610	3055	2	13	y
41	222222	6111111	6611	2255	1	11	y
51	333111	6222000	6620	3155	2	13	y
17	444000	6222000	6620	4055	11	8	y
46	333333	6222222	6622	3355	1	11	y
49	444444	6333333	6633	4455	1	11	y
7	777000	6555000	6650	7055	9	6	y
Superfamily 4.1: $m = 4, s = 1, n = -1, p = 2. 4(w + x) = 4(y + z) - 6 + 10$							
	$wwwwxxxx6$	$yyyyzzzz55$	$6yz$	$wx5$			
20	111100006	0000000055	600	105	19	8	n
23	222200006	1111000055	610	205	15	8	y
26	333300006	2222000055	620	305	16	8	y
30	444400006	3333000055	630	405	27	8	y
4	555500006	4444000055	640	505	25	7	y
42	222244446	5555000055	650	425	1	11	n
44	444444446	5555222255	652	445	1	15	y
8	777700006	6666000055	660	705	6	8	y
Superfamily 4.2: $m = 4, s = 0, n = -5, p = 2. 4(w + x) = 4(y + z) - 30 + 10$							
	$wwww66666$	$yyyyzzzz55$	yz	$w5$			
36	222266666	3333444455	34	25	2	9	y
37	555566666	3333777755	37	55	2	9	y

Table 2. Continued

Superfamily 5.1: $m = 5$, $s = 1$, $n = 0$, $p = 1$; $5(w + x) = 5(y + z) + 30$							
	$xxxxx$	$yyyyy5$	$6y$	$x5$			
21	11111	000005	60	15	13	9	*n*
24	22222	111115	61	25	10	9	*y*
27	33333	222225	62	35	11	9	*y*
31	44444	333335	63	45	8	9	*y*
5	55555	444445	64	55	11	9	*y*
For $p = 6$:							
	$wwwwwxxxxx$	$yyyyy555555$	$6y$	wx			
43	2222244444	00000555555	60	24	3	12	*n*
50	3333333333	00000555555	60	33	1	13	*n*
45	4444444444	22222555555	62	44	2	12	*y*
35	3333377777	44444555555	64	37	1	12	*y*
53	4444477777	55555555555	65	47	1	13	*y*

To convert changes in functional groups to steps we used Fig. 5. We assumed that the reactions shown there can occur regardless of the identity of surrounding functional groups. This assumption requires modification for the oxidation of >CH–CH< to >C = C<, which needs an adjacent group to stabilize the intermediate >CH–C(−)<. For example, the carbonyl >CH–CH–CO– can lose a proton to yield the stable intermediate >CH–C(–)–CO–. This constraint should be considered, and was not. It does not affect our conclusions, because in a sugar an adjacent functional group is always available.

To simplify the analysis we neglected steps involving epimerization without other changes in functional groups, and we have not distinguished among stereoisomers. Thus in our diagrams the side of a carbon on which H and OH occur does not specify chirality, contrary to the usual convention. Epimerization and stereoisomers influenced previous searches for an optimal pentose phosphate pathway (Meléndez-Hevia and Torres, 1988; Meléndez-Hevia *et al.*, 1994). Neglecting these factors limits our results to an approximation that gives a lower bound to the optimum network. A stereochemically optimized network can be no smaller than the optimum we find. Note that our primary concern in this work was to develop and illustrate a new method for generating an ensemble of alternative networks, rather than to specify fully the optimal network.

Phosphorylation prevents the diffusion of sugars out of a cell. So, before a changer produced new compounds, the parent compounds were suitably phosphorylated. For example, in a cleavage reaction both ends of the parent compound were phosphorylated. In a transfer reaction the substrates and products were all phosphorylated. Usually the program transferred the portion of a metabolite containing the initial carbon atom, whereas the terminal carbon atom was phosphorylated. How-

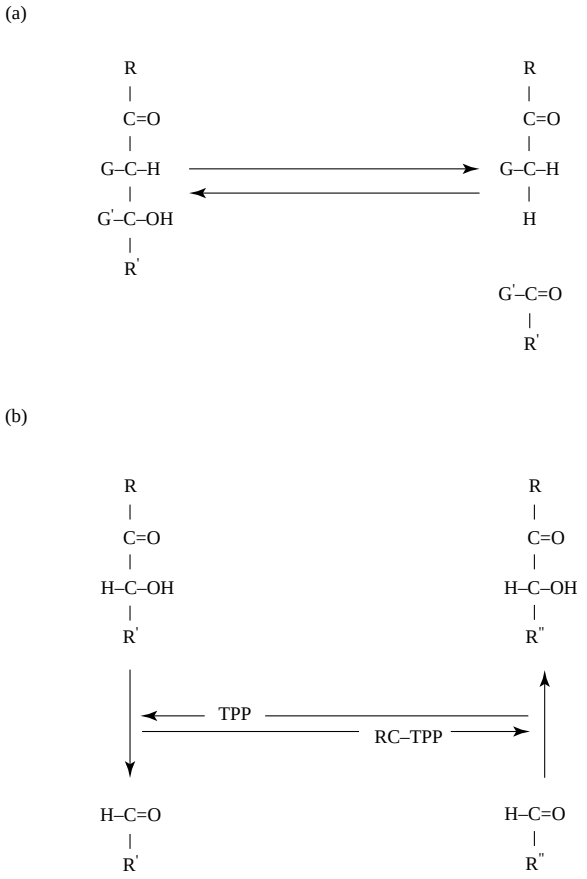


Figure 4. Examples of patterns of functional groups required for a genzyme. (a), Lyase that adds or removes two or more carbon atoms. Aldolase is a real enzyme in this class. R, R' may contain carbon; G, G' do not. (b), Transferase that transfers one or more carbon atoms. Transketolase is a real enzyme in this class. R, R', and R'' may contain carbon. TPP is thiamine pyrophosphate.

ever, sometimes fewer steps were needed if the phosphorylated end was transferred to an unphosphorylated moiety of the recipient molecule.

If the program generated a chemically unstable intermediate, such as an alpha dione, changing the order or type of the functional group changes usually avoided that intermediate. To deal with the irreversibility of some reactions would have required different routes for the forward and reverse fluxes through a pathway. However, we did not deal with such situations, because the resulting R-net would require more ATP than the real pathway.

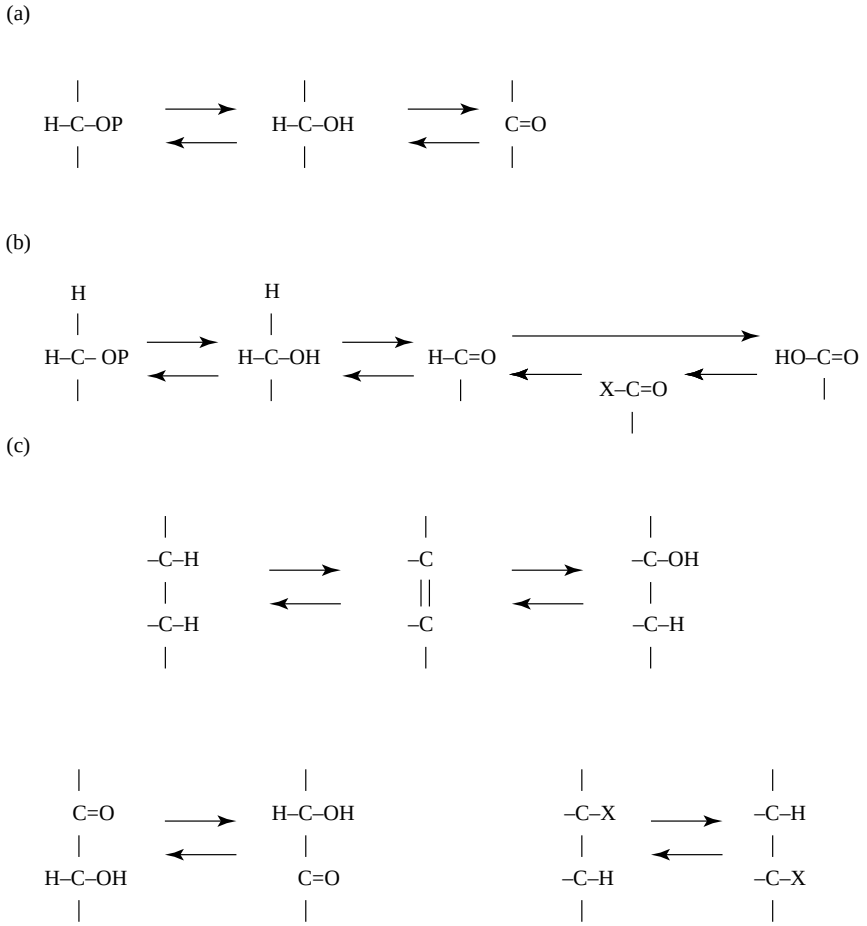


Figure 5. The kinds of steps presumed to underlie changes in functional groups. These steps are used to count the number of enzyme-catalyzed steps in R-nets. Each arrow represents a step. Steps are grouped for compactness, not because they are used sequentially. (So, for example, removal of a phosphate group is not necessarily associated with a subsequent oxidation.) Presumably one enzyme can catalyze each of these steps. Rawn (1989) provides examples for all of the steps but the oxidation of an aldehyde to a carboxylic acid in one step. Examples of the latter occur in enzyme nomenclature Class 1, Subclass 2 (EC 1.2.x.y.). (a), Steps that change functional groups within a carbon chain. (b), Steps that change functional groups at one end of a carbon chain. *X* is a leaving group; addition of *X* to a carboxyl group requires ATP and converts the recipient molecule to a form with higher free energy. *X* may be phosphate, diphosphate, adenosine 5′-monophosphate, adenosine 5′-diphosphate, coenzyme A, or other functional groups. (c), Steps that modify functional groups at two adjacent carbons. Coenzyme B_{12} participates in the exchange of *X* and *H*, as in fatty-acid metabolism.

Table 3. Sixteen genzymes (classes of generic enzymes for changers). Classes for the enzymes used as examples are those listed in the University of Minnesota Biocatalysis/Biodegradation Database (http://dragon.labmed.umn.edu/∼lynda/aboutBBD.html). The numbering is arbitrary.

Operation	Number	Genzyme	Example
Remove 1C:			
	1	Oxidoreductase	Isocitrate dehydrogenase
	2	Oxidoreductase	Pyruvate dehydrogenase
	3	Lyase	Pyruvate decarboxylase
Add 1C:			
	4	Ligase	Pyruvate carboxylase
	5	Lyase	RuBisCO
Remove >=2C:			
	6	Lyase	Aldolase
	7	Lyase	Citrate lyase
	8	Lyase	RuBisCO
	9	Hydrolase	Fumarylacetoacetase
	10	Transferase	Thiolase
Add >= 2C:			
	11	Lyase	Aldolase
	12	Transferase	3-Ketoacyl synthase
	13	Transferase	Thiolase
Transfer 1C:			
	14	Transferase	Serine hydroxymethyl tranferase
Transfer >= 1C:			
	15	Transferase	Transketolase
Transfer >= 3C:			
	16	Transferase	Transaldolase

4.2. ***Restrictions on genzymes.*** To implement the above procedure for generating R-nets we next examine constraints on the types and specificity of genzymes. Unless a restriction is stated, we assume that a genzyme can act on an appropriate set of functional groups, in an unbranched carbon skeleton of arbitrary length, even if some of the corresponding enzymes are not presently found in nature. Several of the genzymes for straight-chain compounds are specialized versions of genzymes that can also operate on branched or cyclic compounds. Other genzymes suitable for this problem, not in Table 3, may exist.

Naturally occurring enzymes have restrictions on the lengths of carbon chains on which they can act. Considerations from biochemistry may allow relaxation of some of these restrictions. For example, Meléndez-Hevia *et al*. (1994) pointed out that transaldolase, which normally transfers three or more carbons, could transfer a 1-

carbon (C1) group from glyceraldehyde, a 2-ketotetrose or a 3-keto sugar. However, we neglected these specialized cases and restricted the transaldolase-like genzyme to transferring three or more carbons. The transketolase mechanism is used in nature to transfer two or more carbons, but it could transfer a C1 group (Meléndez-Hevia *et al*., 1994). Therefore we admitted C1 transfers for the transketolase-like genzyme.

To make our survey more manageable, we reduced the number of R-nets generated from each C-net by seeking criteria for ignoring some of the 16 genzymes in Table 3. By the following reasoning we disallowed use of genzymes 1–5. Genzymes 4 and 5, which add one carbon, need ATP and so would be energetically disadvantageous relative to the real pathway. Ligases that use biotin (genzyme 4) require ATP to form carboxyphosphate, which transfers the carbon to carboxybiotin. A lyase analogous to RuBisCO (genzyme 5) might add carbon dioxide to the end of an aldose, if there is a phosphate group adjacent to the terminal carbon. In this case ATP would be needed to phosphorylate the subterminal carbon. [This is skeleton extension alpha to a carbonyl; see Hendrickson (1970), pp. 472 and 487 and table p. 556.] As it is unclear whether a hydroxyl group could be used in place of the phosphate group, we have neglected this possibility.

All carbons must come from F6P and appear in R5P, or vice versa. So, because genzymes 4 and 5 are not allowed to add one carbon, genzymes 1, 2 and 3 that remove one carbon must also be disallowed. The C-nets that we retained do not have any changers that add or remove a C1—changers 1, 2, 4, 6, 8, 14, and 23 and their reverses in Table 1.

What about transfers of one or more carbons between compounds? Many changers can operate through transfers of two fragments of different lengths. For example, changer 11, $34 \rightarrow 25$, can proceed through transfer of either one carbon or two. Regarding mechanisms of C1 transfer, Meléndez-Hevia *et al*. (1994) showed that an enzyme such as serine hydroxymethyl transferase that uses tetrahydrofolate can not transfer one carbon between aldoses; thus, genzyme 14 can be ignored. However, as discussed above, it is chemically feasible for a transketolase to perform a C1 transfer, though the real transketolase normally transfers two or more carbons. Genzyme 15 was called TK_1 by Meléndez-Hevia *et al*. (1994) to distinguish it from TK_2, the real transketolase. This distinction figures prominently below.

4.3. ***Characteristics of R-nets.*** There were 71 C-nets with five to eight steps and without recurrents other than C5 or C6. Table 4 summarizes information about these C-nets and the associated R-nets. The 71 C-nets are derived from 37 34-vectors, 28 of which have alternative modularizations as pairs or triples of C-nets. We found R-nets as described above for 23 of the 71 C-nets, and recorded in Table 4 the range in the number of functional group changes, and the minimum number of steps, in the R-nets for each C-net. Among the C-nets converted to R-nets there were 13 unique modularizations. The other 10 C-nets are five pairs of distinct modularizations for five 34-vectors. For each C-net of a pair the minimum number of steps in an R-net

was the same; it is unclear whether this is necessarily so in general.

Table 4 shows that larger C-nets (with more steps) tend to generate larger R-nets, but a C-net of given size has a range of R-net sizes. This observation supports our not examining R-nets for C-nets with nine or more steps, since these R-nets are unlikely to be smaller than the R-nets for smaller C-nets. However, in some cases a C-net has an R-net with fewer steps than the smallest R-net generated from a smaller C-net. (The real C-net, with seven steps, has an R-net with 13 steps; for several C-nets with six steps, the smallest R-net has 14 or more steps.) Thus we are not certain that a C-net with nine or more steps could not generate a smaller R-net than the smaller C-nets did. Furthermore, we did not find R-nets for the majority of the C-nets with five to eight steps; our resources were insufficient for this task. Hence we are not certain that the real R-net has fewer steps than any other R-net derivable from the 17 C-nets. However, the following considerations strongly suggest that the real R-net is optimal in this respect.

4.3.1. *R-nets that require use of TK_1.* Of the 71 C-nets, 68 require transfer of one carbon and use of TK_1. Table 4 shows that several of these C-nets have only five or six changer steps, whereas the real pathway has seven. Among these smaller C-nets, several generate R-nets with fewer functional group changes and fewer steps than the real R-net. Meléndez-Hevia *et al*. (1994) closely examined one of these C-nets, with five steps (C-net 1-1 in Table 4). They showed that its best R-net (to be called R-net 1-1 below) seems to have fewer steps than the real pathway. (Ignoring stereochemistry, R-net 1-1 requires 11 steps, and the real pathway requires 13.) However, they showed the side reactions that TK_1 necessarily catalyzes, and the manner in which these must be incorporated in R-net 1-1. Modification of this R-net to include the side reactions gives an R-net with the same number of steps as the real pathway, but with different reactions in the two modules. These asymmetric modules have two or more reactions competing for the same metabolite. The side reactions siphon off TK_1 and intermediate metabolites, reducing the flux through R-net 1-1 below that of the real pathway. The specificity of enzymes in the real pathway does not permit such side reactions.

This argument can be generalized in the following way. Any C-net using TK_1 must have analogous side reactions, reducing the flux through its R-nets. This is so because TK_1 can use any aldose with its C2 in the *R* configuration as a substrate. Thus TK_1 will consume aldoses that are intermediate metabolites in an R-net of interest, degrading its performance. It will produce several aldoses not involved in any particular R-net, and will tend to equilibrate their concentrations. These side reactions compete for TK_1 with the R-net. Thus side reactions reduce the flux through the R-net by reducing the available concentrations of intermediate metabolites and of TK_1.

We suggest that the best R-nets derived from the C-nets that use TK_1 are no better than R-net 1-1, which is worse than the real pathway. If so, the real pathway is better than any of these R-nets. Among the R-nets we generated, the number of side

reactions associated with any R-net that uses TK_1 is at least as large as the number that Meléndez-Hevia *et al*. (1994) found for R-net 1-1, because TK_1 can transfer a C1 among all aldoses. Table 4 shows that the number of steps in each of the best R-nets is at least as large as the number of steps in R-net 1-1. This R-net is worse than the real R-net once the side reactions are included. Therefore, all of the best R-nets using TK_1 are likely to be worse than the real pathway.

An R-net requiring TK_1 is also unlikely to be optimal because many of these R-nets use the same enzymes. So, they can operate simultaneously, competing for enzymes and thereby reducing the flux through each R-net. Two networks (C-nets or R-nets) will be called degenerate if they use the same set of enzymes. In all of those C-nets that do not require addition or removal of one carbon, TK_1 and aldolase (AL) suffice for all the changer steps. This is so because TK_1 can transfer any number of carbons, and AL can remove or add a compound with two or more carbons. Thus in a cell, limited quantities of TK_1 and AL enzymes would be partitioned among many degenerate C-nets. The R-nets derived from these C-nets could also be degenerate, because in connector steps they share enzymes that create, destroy, and rearrange carbonyl and hydroxyl groups. This degeneracy would degrade the performance of R-nets containing TK_1 even more than Meléndez-Hevia *et al*. (1994) considered. It seems unlikely (but not yet demonstrated) that these R-nets collectively would perform better than the R-nets not requiring TK_1, which we now consider.

4.3.2. *R-nets that do not require use of* TK_1. Of the 71 C-nets under consideration, the remaining three do not require transfer of one carbon and use of TK_1. One of these C-nets gives the real pathway; the other two are alternative modularizations of a 34-vector with eight changers. In Table 4 the real C-net, with seven steps, is 1-16; the two eight-step C-nets are 9-10, which has two modules, and 25-7, with three modules. We only constructed the R-nets for the two-module modularization because our earlier work suggested the three-module modularization would have a similar number of steps. The real R-net is the best R-net (has the minimum number of steps) of the R-nets derived from the real C-net; it has 13 steps (recall, we are neglecting the four epimerization steps that actually occur). Table 4 shows that the number of steps in the real R-net is anomalously low, relative to the pattern in which a larger C-net tends to have a larger best R-net. This anomaly occurs because the real R-net avoids connector steps—isomerizations, oxidations, and reductions—that increase the number of steps in other R-nets. The best R-net derived from the eight-step C-net with two modules has 20 steps. Thus the real pathway is likely to be optimal among the best R-nets from the 71 C-nets.

In the real C-net (Fig. 1), AL catalyzes the remainder, with changer $6 \rightarrow 33$. The three changers in the module use two enzymes: changers $36 \rightarrow 45$ and $37 \rightarrow 55$ use TK_2, and changer $46 \rightarrow 37$ uses transaldolase (TA). The remainder has three steps (phosphorylation, changer, isomerization), and each module has five steps (three changers, two isomerizations), so the R-net has 13 steps. Also C-net 1-16 gives an R-net with 17 steps. It has the same three steps in the remainder but uses

TK_2 for all three changers in the module, which has seven steps. The extra two steps in the module, relative to the real pathway, are isomerizations between a 2-keto and a 3-keto sugar. Such an isomerase would be undesirable, since it would catalyze side reactions producing 3-keto sugars from the several 2-keto sugars in a cell. Thus the R-net with TK_2 for all three changers in the module is at a disadvantage relative to the real pathway, regarding the number of steps and the need for an undesirable isomerase.

The eight-step C-net with two modules, shown in Fig. 6(a), has two steps in the remainder and three in each module. In contrast to the real pathway, where the remainder provides inputs (C3s) to the modules, in the eight-step C-net the two modules provide inputs (C2s) to the remainder. Two of the F6Ps must be phosphorylated to fructose 1,6-*bis*phosphate to start the modules. Thus all R-nets generated from this C-net will be more costly to run in terms of energy than the real pathway, which requires only one phosphorylation to start the remainder.

The eight-step C-net with two modules generates 90 R-nets; all of these have more functional group changes than the real pathway. Although this C-net does not require TK_1, some assignments of genzymes to it include TK_1. In particular, the R-net with fewest functional group changes, 31, uses TK_1. Four other R-nets use AL and some combination of TK_2 and TA for the changer steps. All of these four R-nets require conversion of 2-keto to 3-keto sugars, using the kind of undesirable isomerase mentioned above. These R-nets have 36, 39, 42 and 45 functional group changes, or 20, 22, 25 and 28 steps, respectively. The more often TK_2 is used rather than TA in these R-nets, the more connector steps the R-net has. The three R-nets that use AL, TK_2 and TA are degenerate. One of these is the most favorable of the four R-nets; it is shown in Fig. 6(b).

4.4. ***An evolutionary scenario.*** The ensemble of alternative pathways suggests a scenario for the evolution of the real pathway. Our inquiry shows that thousands of R-nets could interconvert 5 C6 and 6 C5. How did organisms evolve the real pathway? The preceding considerations suggest an evolutionary route through which a large initial set of pathways could give access to the real pathway through a few mutations. All of the C-nets where a C1 is not added or removed (as opposed to transfer of a C1 between compounds) can run using TK_1 and Al, because TK_1 can do any transfer, and AL can remove or add a compound with two or more carbons. So, these enzymes can catalyze a large initial set of pathways, many of which have side reactions and degeneracy. Mutation of TK_1 to TK_2 increases the specificity of the transketolase reaction. Nets that use TK_2 but not TK_1 would not have the side reactions associated with TK_1. Such nets include the real C-net, two eight-step C-nets, and perhaps other C-nets with more than eight steps. Our comparison of the best R-nets from the real C-net and the eight-step C-nets suggests that among the C-nets that use TK_2 rather than TK_1, R-nets derived from the real C-net are probably optimal, and so would be favored in selection. However, an R-net that was derived from the real C-net, and that used TK_2 rather than TA, is less satisfactory than

the real R-net. As discussed above, the R-net without TA requires isomerization between 2-keto and 3-keto sugars, with accompanying side reactions.

Use of TA rather than TK_2 for one step in the real pentose phosphate pathway would eliminate these isomerizations and side reactions. TA could have evolved from a precursor of AL: Jia *et al*. (1996) showed that the three-dimensional structures of subunits of TA and AL are quite similar, and some active-site residues and the phosphate-binding site are conserved, despite dissimilarity of their overall amino-acid sequences. These authors suggest that the TA and AL families evolved from a common ancestor.

This scenario has considered selection pressures on the changers. Other selection pressures, operating on connectors, also favor convergence to the real pathway. Selection favors choices of functional groups that reduce the number of steps required to change one functional group to another, so that connectors tend to use fewer steps. Our comparison of the various R-nets derivable from a C-net shows that the best R-nets have fewer steps per functional group change and per connector. (Recall, a connector may include several functional group changes.)

5. DISCUSSION

5.1. ***Our method and its relation to other methods.*** In designing metabolic networks we have transformed carbon skeletons before determining functional groups. This separation is possible because the conservation of carbon atoms is dissociable from the conservation of other atoms in functional groups. The separation simplifies the process of design, because reactions that modify carbon skeletons are often a small fraction of all the reactions in a network. Thus the C-nets, which include only the skeleton-changing reactions, are fewer and smaller than the R-nets, which also modify functional groups.

We looked for optimal R-nets by sequential elaboration and pruning of ensembles of networks, starting from C-nets. In designing C-nets we solved stoichiometric equations for 34-vectors, avoiding futile cycles by a subsidiary optimization. Even so, many 34-vectors were compatible with several C-nets; we selected maximally modular nets, which tend to minimize the number of kinds of enzymes used. Then we added functional groups to C-nets to make R-nets, starting with the C-nets having the fewest steps. In the first stage of this process we assigned 1 of 16 classes of generic enzymes (genzymes) to each changer step in a C-net. Taking into account the functional groups on key metabolites and the functional groups needed for each combination of genzymes, we assigned the remaining functional groups so they change minimally in the connector steps. We examined the R-nets with fewest functional group changes, by hand adjusted unrealistic chemical features, and determined the steps needed to mediate each change. We could then sum the changer steps in a C-net with the connector steps in an R-net derived from it, to find the total number of steps, our index of optimality. Further design stages, with

Table 4. Seventy-one C-nets with five to eight steps without recurrents other than C5 or C6. These C-nets are derived from 37 34-vectors. For each C-net the number of changer steps in the C-net is given first, followed by the superfamily and the family in which it occurs. Under superfamily, the first digit indicates the number of modules in the C-net. Under family, the second index is an arbitrary designation of a C-net within the family. The next column, Remarks, indicates recurrents. A C-net labeled 5 has a recurrent C5 but does not need an exogenous C5 ('starter') to operate, whereas a C-net labeled 5′ needs at lease one starter C5; 6 and 6′ have analogous meanings. Under remarks, the same letter labels pairs or triples of C-nets that are alternative modularizations of the same 34-vector. ** designates the C-net for the real pathway. Next, the changers used in the remainder and the module are listed. Different assignments of genzymes to a C-net give R-nets with a range in the number of functional group changes. These numbers are calculated by the program implementing Appendix 3. They do not include corrections that would be made in adjusting these R-nets by hand, as described in the text. Using Fig. 5 we found the R-net with the minimal number of steps; this number is listed in the last column.

	C-net			Changers used		Range of functional	Minimum steps
		Family	Remarks	Remainder	Module	group changes	in R-nets
5	2.1,	1-1		−16	−21, −22	16–20	11
5	2.1,	2-3		−19, −24, 30	−22	15–19	13
5	2.1,	2-4		−21, −25, 30	−22	18–20	11
5	2.1,	6-1		−22, −25, −33	30	21–23	13
6	2.1,	1-3		−7, −15	−21 −22	24–31	14
6	2.1,	1-5	5′	−9, −20	−21, −22	21–29	12
6	2.1,	2-7	5′a	−9, 19, −21, −22	−22	22–26	15
6	3.1,	3-5	5′a	−9, −19, −21	−22		
6	2.1,	2-8	5′b	−19, −22, −24, 29	−22		
6	3.1,	3-6	5′b	−19, −24, 29	−22		
6	2.1,	2-9	5′c	−21, −22, −25, 29	−22		
6	3.1,	3-7	5′c	−21, −25, 29	−22		
6	2.1,	2-10	d	−13, −22, −25, 30	−22	21–27	14
6	3.1,	3-8	d	−13, −25, 30	−22	19–25	14
6	2.1,	2-11	5′e	−15, −22, 29, −32	−22		
6	3.1,	3-9	5′e	−15, 29, −32	−22		
6	2.1,	6-3	6f	−25, −29, 30, −33	30		
6	3.2,	7-1	6f	−25, −29, −33	30	29–38	18
6	2.2,	18-1	6	−25, −28	−22, 30	23–30	14
7	2.1,	1-7	5′g	−16	−13, −22, −22		
7	3.1,	3-20	5′g	−13, −13, −16, −22	−22		
7	4.1,	4-1	5′g	−13, −13, −16	−22		
7	2.1,	1-11	5′h	−7, −9, −21	−21, −22		
7	3.1,	29-4	5′h	−7, −9, −22, −22	−21		
7	2.1,	1-12	5′i	−9, −11, −22	−21, −22		
7	3.1,	3-14	5′i	−9, −11, −21, −21	−22		
7	2.1,	1-15	5′j	−22, −25, 33	−21, −22		
7	3.1,	3-18	5′j	−21, −21, −25, 33	−22		
7	2.1,	1-16	**	−16	−21, 28, −33	21–35	13
7	2.1,	1-17	5′	−16	−22, −29, −33		
7	2.1,	1-18	6k	21, −25, 30	30, −33	35–39	21
7	3.2,	7-6	6k	21, −25, −33, −33	30	27–31	21
7	2.1,	2-18	6′l	−19, −24, 30	−29, 30	35–52	19
7	3.2,	7-3	6′l	−19, −24, −29, −29	30	29–34	19
7	2.1,	2-19	6′m	−21, −25, 30	−29, 30	29–45	19
7	3.2,	7-4	6′m	−21, −25, −29, −29	30	29–35	19
7	2.1,	2-20	6′n	−15, 30, −32	−29, 30	32–53	20
7	3.2,	7-5	6′n	−15, −29, −29, −32	30	31–39	20
7	2.1,	6-6	5′o	−22, −25, −33	−22, 29		
7	3.1,	3-19	5′o	−25, 29, 29, −33	−22		
7	2.1,	32-3	5′p	−9, −13, −19	−22, −22		
7	3.1,	3-13	5′p	−9, −13, −19, −22	−22		
7	4.1,	4-4	5′p	−9, −13, −19	−22		
7	3.1,	3-12	5′q	−9, 11, −19, −19	−22		
7	2.1,	16-4	5′q	−9, 11, −22	−19, −22		
7	3.1,	3-16	5′r	−13, −25, 29	−22, −22		
7	4.1,	4-5	5′r	−13, −25, 29	−22		
7	3.1,	3-17	5′s	−24, 29, 29, −32	−22		
7	2.1,	6-4	5′s	−22, −24, −32	−22, 29		

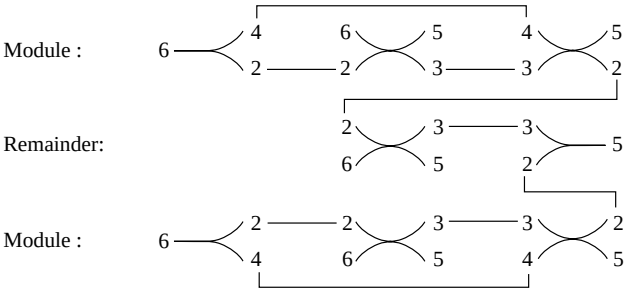
	C-net			Changers used		Range of functional group changes	Minimum steps in R-nets
		Family	Remarks	Remainder	Module		
8	2.1,	1-26	$5't$	$-7, -15$	$-13, -22, -22$		
8	4.1,	4-9	$5't$	$-7, -13, -13, -15$	-22		
8	3.1,	3-26	$5't$	$-7, -15$	$13, -22, 22$		
8	2.1,	1-28	$5'u$	$-9, -20$	$-13, -22, -22$		
8	4.1,	4-14	$5'u$	$-9, -13, -13, -20$	-22		
8	3.1,	3-28	$5'u$	$-9, -20$	$-13, -22, -22$		
8	2.1,	2-25	$6'v$	$-15, -19, -29, 30$	$-29, 30$		
8	3.1,	3-33	$6'v$	$-15, -19$	$-29, 30$	37–59	19
8	2.1,	9-10	w	$9, -20$	$11, -15, -20$	31–60	20
8	3.1,	25-7	w	$9, 11, 11, -15, -15$	-20		
8	2.1,	9-18	$5'x$	$5, -22$	$-9, -21, -22$		
8	3.1,	3-30	$5'x$	$5, -9, -9-21, -21$	-22		
8	2.1,	9-20	$5'y$	$5, -22$	$-22, -24, 29$		
8	3.1,	3-31	$5'y$	$5, -24, -24, 29, 29$	-22		
8	2.2,	18-3	$5'z$	$-25, -28$	$-22, -22, -29$		
8	3.1,	3-32	$5'z$	$-25, -28$	$-22, -22, 29$		
8	4.1,	4-16	$5'z$	$-25, -28, 29, 29$	-22		
8	2.2,	18-4	$6'a'$	$-25, -28$	$-29, 30, 30$		
8	4.1,	8-1	$6'a'$	$-25, -28, -29, -29$	30	34–53	22
8	3.2,	7-8	$6'a'$	$-25, -28, -29, -29, 30$	30		
8	2.2,	33-1	$5'b'$	$-9, 20$	$-19, -22, -22$		
8	3.1,	3-29	$5'b'$	$-9, 20$	$-19, -22, -22$		
8	4.1,	4-15	$5'b'$	$-9, -19, -19, 20$	-22		

elaboration and pruning beyond the scope of our work, would include adjusting for stereochemistry and for special features of specificity and mechanism of enzymes; designing regulation, and modeling dynamics. Taken together, the stages in our approach could be integrated into one computer program. Given the task, the list of changers, and the genzymes, the program would compute the R-nets having the fewest steps.

Our approach generalizes previous strategies for designing metabolic networks with specified inputs and outputs. Previous approaches (Seressiotis and Bailey, 1988; Happel *et al*., 1990; Mavrovouniotis *et al*., 1990; Meléndez-Hevia, 1990; Nuño *et al*., 1997) designed networks using a specific set of metabolites and enzymes. While these approaches can generate networks efficiently if this set is small, our method surveys a much larger set. The survey is feasible because our method relies on the generic properties of metabolites and enzymes, rather than using their specific identities and affinities, to do a sequence of elaborations and prunings. More specifically, at the beginning of our procedure the only inputs are the key metabolites, the types of admissible carbon skeletons and transformations among them, and the generic enzymes available. The first step, generation of C-nets, depends only on the stoichiometry for transforming carbon skeletons of key metabolites, and on changer reactions not identified with enzymes. In the following steps one assigns functional groups to carbon skeletons. We sought R-nets with few steps, and imposed this constraint by adding functional groups first to smaller C-nets. Other search algorithms might generate smaller C-nets earlier. In the first stage of adding functional groups we assigned genzymes to changers, rather than using enzymes that mediate specific reactions. This leaves the assignment of functional groups not associated with genzymes to a later stage. Limitations on the identities of metabolites are only imposed late in the determination of functional groups.

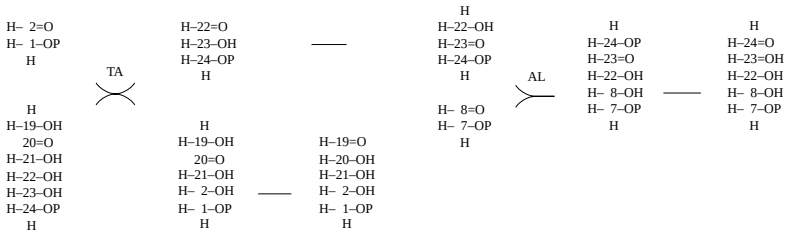
At each stage the search generates complete networks, rather than fragments of networks that are finally assembled into complete networks. Iterative elaboration and pruning progressively increase the detail with which the complete networks are

(a)



(b)

Remainder:



Module:

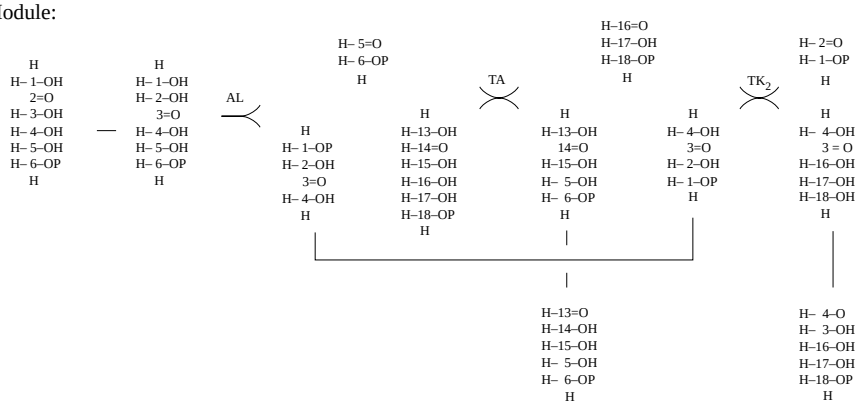


Figure 6. (a), The eight-step C-net with two modules that does not require transfer of one carbon by transketolase. (b), The remainder and one module from an R-net derived from this C-net. The R-net uses the genzymes aldolase (AL), transaldolase (TA), and the transketolase that can transfer two or more carbons (TK_2) for changer steps. Lines without labels represent connector steps. This R-net has 36 functional group changes and 20 steps; the remainder has six steps and each module has seven steps. (Note that in the remainder the last transformation represents two steps, an isomerization and a dephosphorylation. The first step in the module also represents two steps, an isomerization and a phosphorylation.) The numbers that label carbon atoms were assigned during computation. Because we ignored the asymmetry of carbon atoms, the placement of all hydroxyl groups to the right of each carbon skeleton is an arbitrary convention. –OP represents a phosphate group.

specified. Pruning can not only restrict the search, but also can direct it toward a desired class of alternatives.

The work of Nuño *et al*. (1997) resembles our approach in several ways, but differs significantly. These authors pointed out that the pentose phosphate pathway is one of many routes through a network of reactions that interconvert monosaccharides. In analyzing this network they noted the distinction between changer and connector reactions, and they constructed C-nets. The enzymes in their changer reactions were more specific than our genzymes; they assumed that TK and TA can work on monosaccharides of various chain lengths, as we did, but that TK always transfers a 2-carbon fragment and TA transfers a 3-carbon fragment. We allowed the corresponding genzymes to transfer longer fragments.

Nuño *et al*. (1997) offered a way to solve for all flux distributions that are compatible with any of the global stoichiometries that can be imposed on the network. The flux distributions for one stoichiometry correspond to the ensemble of pathways that we found. They found a basis set of independent flux distributions, from which all other flux distributions can be constructed as linear combinations. Several of the fundamental flux distributions in the basis set are futile cycles; so, futile cycles need not be regarded as an irrelevant nuisance. Evidently there are various routes to constructing an ensemble of networks.

5.2. ***Extension of our approach to analysis of other networks.*** Our approach could be extended to more complex metabolic networks. A potential difficulty is that the procedure we used to avoid futile cycles may be harder to apply as the discrepancy increases between the minimal C-net that can accomplish a task and the actual C-net that does accomplish it. This is, our algorithm may not discriminate between a small C-net with futile cycles and a larger C-net without futile cycles. An alternative approach that might overcome this difficulty is to generate C-nets using an algorithm such as Mavrovouniotis *et al*. (1990) presented, building up pathways sequentially from the list of changer reactions.

We have dealt with straight-chain carbon compounds. To include branching or cyclic carbon skeletons in the design of C-nets, it might seem that one could generate C-nets in two stages. The first stage would consider only the number of carbon atoms in each skeleton, as done here. Each of these C-nets could then be elaborated into a set of alternatives by allowing each number of carbon atoms to represent one of several alternative skeletons, and specifying changer reactions that interconvert these skeletons. However, there is a flaw in this approach; a futile cycle in a network where a number of carbon atoms represents each skeleton is not necessarily a futile cycle when the skeletons themselves are used. Alternatively, one could proceed as we did, but use changers that acknowledge differences among carbon skeletons with the same number of carbon atoms. The assignment of genzymes to C-nets and the specification of functional groups would then proceed as in our work.

Generating an ensemble of C-nets produces a combinatorial explosion. Most of these nets can be pruned by invoking constraints that eliminate unacceptable candi-

dates. We did this with C-nets to avoid futile cycles. Limitations on resources—the number of steps, the number of kinds of genzymes—are especially useful for pruning (Baskin *et al*., 1992; Mittenthal *et al*., 1993). Resource limitations select for increased structure; in particular, repeated use of a module—a cluster of processes with few inputs—is helpful. Maximal modularization—including the maximum number of processes within repeated modules—reduces the demand on resources.

Hindsight shows that in applying our method, restricting the set of genzymes is crucial to limiting the ensemble of C-nets. If possible, only genzymes without undesirable side reactions should be used. (Following this dictum, we would have excluded TK_1.) The set of genzymes limits the set of changer reactions available for use. The set of changers limits the ensemble of C-nets that will be constructed.

The generation of R-nets from C-nets also produces a combinatorial explosion. This explosion was acceptable for the few and small C-nets that we considered. However, to deal with more or larger C-nets we suggest invoking a second, subsidiary criterion of optimality: ensure that analogous compounds in two instances of the same module are the same, by searching among solutions of a pair of equations rather than conducting an exhaustive search among a much larger set of alternatives. Appendix 3 explains this procedure.

Aspects of our strategy may carry over from metabolism to the design of other intracellular networks (gene regulatory, cytoplasmic signaling) and multicellular networks (neural, endocrine, immune). We constructed a sequence of ensembles of nets. The nets in an ensemble are generated by forming all of the allowed combinations of a set of units. The units are abstract, generic entities that are made more specific in later ensembles. The process of abstraction neglects differences among units, focusing on their common attributes. For example, diverse metabolites with the same carbon skeleton could undergo a changer reaction. We used classes of enzymes that could catalyze the same reaction in diverse molecules. For changers we called these genzymes; for connectors, we assumed that there was a set of enzymes that could change any functional group to any other. At a higher level, a module can be regarded as an abstraction that neglects differences among units. Different forms of a module have as common attributes the same inputs and outputs, but differ in internal organization.

5.3. ***Conclusions: evolution, architecture, optimality.*** One can study the design of a network to get insights into its evolution, architecture, and optimality. As regards evolution, the path through which we generated realistic networks—from networks of generic units with broad input–output relations to networks of specific units—is a path that diverse networks may have followed in evolution (de Duve, 1991). Our work provides a scenario for the evolution of the pentose phosphate pathway, discussed in Section 4.4.

With respect to architecture, we found that C-nets have characteristic structures—repeated modules, and a remainder (a module that is not repeated). Modularization can occur in various ways, even for the same set of steps. The ensemble of C-

nets inherits a structure from the modularization: the C-nets group in families and superfamilies because some modules or remainders have similar inputs and outputs despite interior differences.

Our work suggests, but does not prove, that the real pentose phosphate pathway is optimal with respect to specific criteria, and within a specified domain. Meléndez-Hevia *et al*. (1994) concluded that the real pathway is optimal within an ensemble of networks that use a small number of realistic enzymes and real metabolites. We inquired whether the domain of optimality extends to a wider variety of enzymes and metabolites. Our optimality criterion combines minimization over the number of steps and the number of ATP molecules used. According to these criteria, all but three of the R-nets that we generated (five-step C-nets 1-1 and 2-4, and six-step C-net 1-5; see Table 4) are inferior to the real one. Meléndez-Hevia *et al*. (1994) used arguments about side reactions and kinetic simulations to show that an R-net with non-identical modules, derived from C-net 1-1, is inferior to the real R-net. Analogous considerations would be needed to exclude other alternatives. These considerations are beyond the scope of our work, which had as its primary objective the demonstration of a new method for constructing metabolic networks. We did not evaluate the number of steps and ATP molecules used for many of the R-nets corresponding to the C-nets that we generated. We have not investigated consequences of non-identical modules, or of allowing more than four kinds of changers.

It is unclear whether an unequivocal decision about the optimality of the pentose phosphate pathway under physiological conditions is possible. As Meléndez-Hevia *et al*. (1994) pointed out, the number of TK_1-mediated side reactions that must be considered depends on whether the pathway is running from C6s and C5s or vice versa. The relative extent of its operation in the two directions must vary with conditions in a cell, so the number of steps is ambiguous. (The phosphorylation needed in going from C6s to C5s is not needed in the reverse direction.) The number of steps also depends on which C6 and C5 one takes as the endpoints of the pathway. Glucose or fructose might be taken as the C6 sugars, and the possibilities for C5 sugars include ribose and ribulose. We chose F6P and R5P as endpoints, and compared R-nets on this basis; but different choices of endpoints can give different numbers of steps in networks being compared. Furthermore, the pathway may use diverse C6s and C5s, in time-varying proportions. Even if a complicated optimization function included the number of steps, kinetics, ATP usage, redox reactions, side reactions, and degeneracy, the unpredictable temporal variation in the metabolic constraints on a cell might prevent a formal determination whether the pentose phosphate pathway is optimal in a natural environment. Indeed, it is possible in principle that different pathways are optimal in different environments.

Nevertheless, comparison of the real pathway with alternatives has emphasized its favorable characteristics. It uses no reducing or oxidizing compounds. It requires only one ATP in one direction of flux in the pathway; the other reactions are reversible. It lacks the side reactions and degeneracy associated with many

alternative pathways. Its operation does not depend on recurrent inputs; hence it is relatively insensitive to concentrations of metabolites other than its key metabolites, cofactors, and currency molecules. These attributes would give a real pathway a selective advantage in competition with networks that need more currency molecules, have more irreversible reactions, have side reactions or degeneracy, or depend on recurrent inputs. This combination of advantages, more than any one of them, may have favored the ubiquity of the real pathway.

ACKNOWLEDGEMENTS

This work was partially supported by NSERC Operating Grant 5-54891.

APPENDIX 1. GENERATING C-NETS

Suppose we wish to generate a large collection of networks of reactions which can interconvert two key metabolites. We abstract this problem by regarding the two key metabolites as two carbon chains Cm and Cn of length m and n respectively. Here, Cm represents m carbon atoms bonded linearly to each other. The maximal number of functional groups is $2(m+1)$, as each of the two terminal carbon atoms can bind at most three functional groups and the $m-2$ intermediate carbon atoms can bind at most two functional groups each.

For the moment, we treat Cm and Cn as carbon chains; functional groups will be specified later.

For nontriviality, we assume n and m are different. Now, a balanced reaction for the carbon skeletons Cm and Cn is

$$nCm \rightarrow mCn \tag{1.1}$$

The coefficients n and m in (1.1) are necessary for the conservation of carbon atoms. In the present example of the pentose phosphate pathway, $n = 6$ and $m = 5$ so Cm represents the carbon skeleton of R5P, and Cn represents the carbon skeleton of F6P. In this case (1.1) is the overall expression for a collection of networks of reactions.

We begin to generate a class of networks by decomposing (1.1) into a sum of bimolecular reactions each of which, like (1.1), is generic. That is, each of the bimolecular reactions that we permit to occur in a representation of (1.1) only specifies lengths of carbon chains; functional groups remain unspecified. In general, (1.1) cannot be decomposed uniquely into a sum of reactions from Table 1 with integer coefficients. Indeed, we exploit this nonuniqueness to generate a large collection of decompositions of the interconversion (1.1) in terms of the changers in Table 1. Each decomposition corresponds to a network of changers summing to give (1.1); we call such networks of changers C-nets.

To express the various decompositions of (1.1) in terms of the reactions in Table 1, let $u = (u_1, \ldots, u_{34})^t$ be a vector with integer entries. The entry u_i represents the number of occurrences of reaction $2i - 1$ minus the number of occurrences of $2i$ from Table 1. Next, we convert the forward reactions in Table 1 to a 7×34 matrix V. Each column of V is a forward reaction from Table 1; each row in V corresponds to the number of carbon atoms in a compound. Thus, the first column of V corresponds to $2C1 \rightarrow C2$ and is $(-2, 1, 0, 0, 0, 0, 0)^t$; the second column of V corresponds to $C1 + C2 \rightarrow C3$ and is $(-1, -1, 1, 0, 0, 0, 0)^t$ and so on up to 34. That is, the entries in column i are the coefficients of Ci in reaction i from Table 1.

Now, the product Vu is the sum of reactions r_i in Table 1 with coefficients u_i. That is, Vu is an overall interconversion of the form (1.1). In this form it is seen that there are diverse choices for u that can give the same interconversion, i.e., in general, it is possible to have u, u' satisfying $Vu = Vu'$ but $u \neq u'$. Let the vector W represent (1.1). We try to solve $W = Vu$ for vectors u. Since V is determined already, the collection of representations of the overall reaction W in terms of reactions from Table 1 is the set $\{u \in \mathbb{Z}^{34} | W = Vu\}$.

Ideally, we would want to solve for u by $u = V^{-1}W$. However, V^{-1} does not typically exist because there is no unique solution. Indeed, it is possible to obtain an infinity of solutions from 'futile cycles' because one can cycle through a sequence of reactions arbitrarily many times without affecting the final product. For instance, the sequence $2C2 \rightarrow C4$, $C4 \rightarrow C3 + C1$, and $C3 + C1 \rightarrow 2C2$ is a futile cycle. A further complication that the procedure must resolve is that solution vectors u must have integer entries.

The procedure we used to obtain solution vectors u is restricted to $\mathbb{Z}^{34}$ and conducts a search over candidate vectors u so as to avoid the nonexistence of V^{-1}. Futile cycles are avoided by a minimization procedure on the entries of u.

Briefly, the procedure we used to obtain a useful class of u-vectors is as follows. We begin by restricting u-vectors to four nonzero entries, because the real pathway only uses four kinds of reactions. This means that, for each choice of four nonzero components of u, we can reduce V to a matrix V' that has four columns. Now, the solutions we generate correspond to C-nets that can accomplish the interconversion (1.1) using at most four of the C-reactions in Table 1, although each of the four reactions may be used several times in either direction. In the form our solutions take, the sum of the absolute values of the entries in the solution u-vectors is the number of bimolecular reactions in the network or networks it leads to. Note that if four C-reactions from Table 1 are allowed there are $C(34, 4) = 46\,376$ combinations.

Suppose four reactions to be assigned nonzero coefficients have been chosen. For the pentose phosphate pathway, we have $W = (0, 0, 0, 0, 6, -5, 0)^t$ and we seek solutions to

$$V'u' = W, \tag{1.2}$$

where V' has the columns of V corresponding to the entries of u that are allowed to be nonzero, and u' is the vector of those nonzero entries. It is seen that V' is 7×4, u' is 4×1 and W is 7×1.

Now consider the rank of V'. If the rank of $V' \leq 3$ then there is either no solution or an infinity of solutions to equation (1.2). In the latter case, one can find a solution which uses only three reactions from Table 1. Thus, we only considered here the cases in which V had full rank 4.

Even when V has full rank, equation (1.2) need not admit a solution. Our program detects this case by recording V' and u' in row-echelon form and solving to get a solution of 0 for a specific u'_i. In the case that a nontrivial solution was found, we still had to search for futile cycles because a nontrivial solution might not be unique. In the case of nonuniqueness, we imposed a rule for selection of a particular solution vector which we hoped would have the sum of the absolute values of the entries small. Specifically, we chose the solution vector u' in which the lowest subscript in u was positive when the entry in W corresponding to an undetermined equation was zero. When the entry in W was not zero, we chose the sign of the highest subscript in u to be the same as the entry in W and to have the lowest absolute value permitting an integer solution.

Coding and implementing this procedure for the pentose phosphate pathway generated 275 solution 34-vectors u. Each 34-vector corresponds to a decomposition of (1.1). Given a 34-vector, it is straightforward to write down the reactions it indicates and connect them so as to form a network accomplishing (1.1). We deal with connectivities of these C-nets next.

APPENDIX 2. MODULARIZING C-NETS

Given a 34-vector obtained from the procedure described in the Appendix 1 it is desirable to list the various network connectivities it permits. Generally, several nonisomorphic connectivities are consistent with a particular u-vector. While it is possible to list all of these, we have sought only those connectivities that have what we call a modular structure.

Here we search systematically for modules by finding collections of reactions that occur at least twice in a C-net, and coupling them later. As a consequence, even though we were only searching for modularizations with repeated copies of a single module, our method occasionally generated a modularization with repeated copies of two distinct modules, because the search initially neglects the coupling.

Our systematic search is based on applying the Euclidean division algorithm to vectors u. We are interested in decomposing networks associated to a u-vector into what we call maximal modularizations of order 1, using maximal modules. To define these terms precisely: a modularization of a network is a partition of its reactions into modules and a remainder; maximal if the reactions in the remainder cannot be used to form another copy of any module that has already been identified;

and is of order 1 if only one module is repeated and the rest of the reactions are in the remainder. Thus, a maximal modularization of order 1 for a u-vector is a representation of the u-vector as a collection of identical modules and a remainder in which one cannot find extra copies of any module. A module itself is maximal if no other module that can appear the same number of times in a network, i.e., in a u-vector, has a larger number of reactions in it.

We search for maximal modularizations of order 1 using maximal modules by using the following proposition. For the convenience of proof we assume the u_is are nonnegative. In practice, the u_is can be negative, indicating the reverse reaction. In such cases, the proposition can be applied to $u^* = (|u_1|, \ldots, |u_{34}|)$ and the sign of u_i is applied to both m_i and r_i defined below. The divisor α is the number of copies of module m and r is the remainder.

PROPOSITION 1. *The collection of maximal modularizations of order 1 of a u-vector $u = (u_1, \ldots, u_{34})$ with $u_i \geq 0$ by maximal modules $m = (m_1, \ldots, m_{34}) \in \mathbb{Z}^{34}$ with $m_i \geq 0$ is a subset of the class of modularizations given by*

$$u = \alpha m + r \tag{2.1}$$

where $r = (r_1, \ldots, r_{34}) \in \mathbb{Z}^{34}$ with $r_i \geq 0$ is the remainder, $\alpha \in \mathbb{N}$ varies over $\{2, \ldots, \max_i u_i\}$ with $r_i \leq \alpha - 1$.

Proof. Suppose m is a maximal module so that α copies of m is a maximal modularization. Then, $u = \alpha m + r$. If $r_i \geq \alpha$ for some i, let e_i be the vector with 1 in the ith position and zeroes elsewhere. Now we have

$$u = \alpha(m + e_i) + (r - \alpha e_i). \tag{2.2}$$

That is, $m + e_i$ is a module which is larger than m and can also appear α times, so m itself is not maximal, a contradiction.

We note that the bounds on α are trivial: $\alpha \geq 2$ is necessary by the definition of a module and $\alpha \leq \max_i u_i$ is necessary since u limits the number of C-reactions that can be used to form modules.

Proposition 1 does not say that every expression of the form (2.1) gives a decomposition of the form we want. It only says that every decomposition of the form we want can be expressed in the form (2.1). Consequently, we use Proposition 1, i.e., the Euclidean division algorithm, to search for maximal modularizations of order 1 using maximal modules.

As each u-vector typically leads to one or more maximal modularizations of order 1 in terms of maximal modules depending on how many values of α are permissible, given u and α one can find both m and r. One can examine each module m that arises for each u-vector and couple the reactions in it. This must be done in exactly the same way for each copy of m so as to ensure that each copy of the module functions identically.

To obtain an overall connectivity for the C-net, we began by choosing a connectivity within modules. The connectivity within a repeated collection of reactions was chosen so that as few reactions as possible would be disconnected from other reactions in the collection. In general, this could not be done uniquely. One connectivity was chosen heuristically based on maximizing similarity of candidate modules where possible. The C-nets from the modules were then connected to form a C-net for the overall reaction by using the reactions in r. Again, in cases where several connectivities were possible, we chose one based on maximal symmetry. This was not done by any formal criterion; it was merely by eye. By the construction of the u-vectors one can always find at least one overall connectivity compatible with the modularization. However, uniqueness is not always possible.

APPENDIX 3. ASSIGNING GENZYMES AND FUNCTIONAL GROUPS TO C-NETS

Once one has a list of basic C-reactions such as those in Table 1, and has solved for modularizations, thereby generating a list of candidate C-nets, one must assign functional groups to the C-skeletons in each C-net so as to obtain completely specified compounds. The assignment of genzymes to changer reactions (those that affect the C-skeletons) effectively specifies some of the functional groups by the requirements of the genzymes. However, other functional groups remain unspecified because they are not affected by the genzyme actions. In addition, we must connect the outputs of some changer reactions to the inputs of other changer reactions, through connectors. Some of the functional groups can be carried forward from output to input; others can be carried forward from the input F6Ps or must be converted to the output R5Ps. To respect the module structure we will have to ensure that all the functional groups in corresponding compounds in all occurrences of a module are the same. This process is somewhat involved, so we begin by choosing a connectivity compatible with a modularization for a C-net and assigning genzymes. Only then do we turn to the remaining functional groups.

Suppose we have chosen a fixed connectivity for a C-net as described at the end of Appendix 2. We begin with generic assignment in which we assign genzymes and those functional groups necessary for genzyme function. First, we compile a list of genzymes that are permitted. This is given in Table 3. Each genzyme requires certain functional groups and can only act on certain C-skeletons. Starting with a C-net as above we assign genzymes in all possible ways and assign the functional groups necessary for their function. That is, for each of the 34 changer reactions in Table 1 we listed the genzymes in Table 3 that can mediate it; Table 1 includes this list. Then, starting with a C-net and its connectivity we choose a reaction in its module and assign the first genzyme that can mediate that reaction to all instances of it in the module. Then we choose the second reaction in the module and assign all occurrences of it in the module to the first genzyme that can

mediate it. Doing this for the third and fourth type of reaction in the module and repeating this procedure for the remainder completes the first genzyme assignment. Recall that each C-net uses four changer reactions, so cycling through all possible genzymes for each changer reaction generates the complete collection of generic assignments. We call assignments homogeneous when all instances of a changer reaction in the C-net are mediated by the same genzyme. Here, we have included all homogeneous assignments, and we have permitted nonhomogeneous assignments in which a reaction in the remainder was assigned a different genzyme from the one it was assigned within a module. Now, we can assume that we have a C-net with specified connectivity and a genzyme assignment. Both of these features respect the module structure. Note that connectors have not yet been specified.

Next we present an algorithm for what we call specific assignment in which we fill in the functional groups that are not specified by the choice of genzymes. The key issue in the algorithm is to produce fully specified nets that have the fewest number of steps and to do so efficiently. In the present case, this can be done exhaustively—it is the brute-force method. Nevertheless, we outline a procedure which can be used to produce the nets with the fewest steps by solving certain equations. We present the algorithm assuming there are no recurrents, but we indicate how recurrents can be dealt with.

Number each compound in the C-net. There are k compounds where

$$k = 11 + \text{ (number of compounds within the C-net);} \tag{3.1}$$

the added 11 counts the 5 F6Ps and 6 R5Ps. Note that the C-net with its specific inputs (F6Ps) and specific outputs (R5Ps) is a collection of changers and connectors. Number all of these reactions in order of use. That is, we start by assuming the C-net is to be used in one direction only which we call forward: the 5 F6Ps are converted to 6 R5Ps. Specifically, label the reactions that must occur as prerequisites for any further reactions by 1, 2 and so on, until all are labeled. Such a group of reactions that are prerequisite for all that follow will be said to occur during a period. Next, look at the reactions occurring during the next period and continue counting until all reactions are numbered. If two reactions occur during the same period, the order in which they are labeled will not matter.

Note that now the connectors which take the F6Ps are not necessarily the first five reactions. One or more—possibly all—F6Ps may enter after the first period. Let l be the sum of three terms:

$$l = \text{ (number of changers) } + 11 + \text{ (number of connections).} \tag{3.2}$$

A connection is a sequence of connector steps that converts F6P, or a compound that is an output of a changer reaction, to an input of another changer reaction, or to R5P. Figure 1(b) illustrates periods and connections. Thus l is the sum of the changers from Table 1 that define the C-net, plus the five connections that join the

F6Ps to the inputs of changers, plus the six connections that join the 6 R5Ps to the outputs of changers, plus the connections internal to the C-net.

The F6Ps may enter in any period, and the R5Ps may be produced in any period past the third. In addition, reactions occurring within a module must be noted as such, so that two instances of the same reaction in different modules will be assigned the same functional groups. We assume that any recurrents have been removed by omitting one connection for each. These must be replaced at the end of the procedure we describe below.

To specify the algorithm, we adopt the following conventions.

CONVENTION A: PERIODS AND LABELING. Compound numbering and reaction numbering must be consistent with period numbering, which is sequential. Apart from the period ordering, compound and reaction numbering is arbitrary.

We adopt the convention that compounds never wait around to be used; they are made just prior to their use. When all compounds prerequisite for the next period are available, the corresponding reactions occur. A benefit of this convention is that all nets will have alternating connections and changers. That is, a collection of connections is performed during the first period, a collection of changers is performed during the second period, a collection of connections occurs during the third and so on: odd steps are connections, even steps are changers [Fig. 1(b)].

CONVENTION B: RECURRENTS. We ignore the exact identity of recurrent compounds, in terms of the functional groups that they bear, to a certain extent: we choose them to be products which are not among the R5Ps that the R-net produces as outputs. We specify separately the functional groups that a recurrent compound must have when it is an output and when it is an input, adding a connector for the appropriate conversions later. Possibly, this will leave some functional groups unspecified—those which are not determined by genzymes, by reactants producing the recurrent or requiring it, or by module structure. The l in (3.2) counts the number of functional-group changes in the C-net apart from connector steps that are necessary to replenish a recurrent compound.

CONVENTION C: PHOSPHORYLATION. We ignore phosphorylation.

CONVENTION D: FUNCTIONAL GROUPS. The activity of a genzyme requires critical functional groups. We assume that genzymes also specify where the carbons in the substrates end up in the products. In changers, noncritical functional groups stay on the same carbon in the products as they were on in the substrates.

CONVENTION E: CONNECTIONS. Modules contain the changers plus the connections between them. In particular, connections leading into or out of a module are not part of the module. We take as our net all the couplings except the connections required to replenish recurrent compounds. We assume that each changer reaction

in period 2 has at least one F6P (possibly modified in a connection) as an input. This ensures we have some starting functional groups to carry forward.

We define a minimal path between two compounds with the same number of carbon atoms as the connection requiring the fewest functional group changes. Each compound is regarded as a matrix in which the number of rows is the number of carbons and the number of columns is equal to three, since the maximal number of functional groups a carbon atom can have when it is in a chain is three. We will evaluate path length between two compounds by comparing their matrices, as shown in Step 5. We consider two approaches. One is to use a distance measure H which counts the number of discrepancies in the entries of the matrices for the compounds. This measure gives a pair of Diophantine equations of which the solutions contain the possible functional groups. This approach using a distance measure reveals that if there are two modules then the optimal R-net specifies compounds that enter the modules as the midpoint between the outputs of the changer reactions that produced them. We have not investigated the case of three or more modules. Alternatively, one can use the brute-force method, evaluating the number of functional-group changes for each possible assignment of functional groups that are not specified by the genzymes or the F6Ps and R5Ps, and keeping the R-net with the smallest number of changes. In dealing with a small number of small nets the brute-force method is easier to code then Step 5 below, and we used it.

Subject to these conventions, we give a step-by-step procedure for generating R-nets. The brute-force method and the distance-measure method are the same up to Step 5.

STEP 1: DIAGRAM OF THE C-NET. We begin by assuming that the compounds have been labeled and the labels entered on a diagram of the C-net, that all connections and changers have been numbered, and that the input F6Ps and output R5Ps have been inserted, with their functional groups, in the proper periods.

STEP 2: TABLE OF THE C-NET. Form a table indexing the carbon atoms of the overall inputs, namely 5 F6Ps, with numbers from 1 to 30. The number of rows in the table is 30; the number of columns is the number of periods plus one. An entry is the index of a carbon atom, with notation of the other carbon atoms and the functional groups to which it is bonded. As we proceed we will fill in entries. Fill in the entries of the table that are specified by the F6Ps and the R5Ps at the appropriate periods. Choose an allowable assignment of genzymes following Convention D. Fill in the entries of the table that are specified by the genzyme assignment. Now, each compound in a C-changing reaction has some of its functional groups specified.

STEP 3: MINIMAL PATH FOR THE FIRST PERIOD. Consider what happens during the first period. Typically, an F6P enters the net; a recurrent may or may not be necessary. Indeed, two or more recurrents or two or more F6Ps may enter the first period. Choose one of these inputs and loop over the rest.

If the input is an F6P then it enters a connection joining it to a changer. The F6P must be converted into a compound that will serve as a substrate of the genzyme assigned to that changer. We change whatever functional groups have to be changed for the genzyme to work. If a functional group is not specified by the genzyme we specify it by the corresponding functional group on the input F6P. This produces a path with the minimal number of functional-group changes from F6P to the input C6. If the input is not an F6P, it is a recurrent, and it enters a changer directly, so some of its functional groups are specified by the genzyme for the changer.

At this point we have not considered the module structure because the minimal path argument will give the same results for corresponding F6Ps entering different modules. If F6Ps enter a module the module structure is preserved because the functional groups are specified by the genzymes, which respect the module structure. Also, recurrents that enter a module in the first period are only (at this point) affected by the genzymes, which respect the module structure. Now we have carried the information from the inputs to period 1 to outputs of that period.

STEP 4: MINIMAL PATHS FOR THE SECOND PERIOD. Next we use the outputs from period 1 as the inputs to the next reaction. The outputs from period 1 come from a connection and enter a changer. Loop over all compounds produced during that period. Choose one to start; it is either a recurrent or it is fully specified from Step 3. If it is a recurrent, it is partially specified by the genzyme and we leave it alone; its unspecified functional groups stay unspecified except for the module structure, which we address later. If it is fully specified there is no problem. By Convention D, carry the functional groups on the compound to the corresponding functional groups on the output(s) of the reaction at period 2.

STEP 5: MINIMAL PATHS TO AND FROM MODULES. At the end of period 2 we have compounds from the previous changers, and we put them through a connection. Note that F6Ps can enter the third period. We attach them by a minimal connection to the input C6s. If we have an output C5 we attach it to R5Ps by a minimal connection. Other compounds that enter the third period had to have been produced by an earlier changer and must go through a connection to enter the next changer.

There are several cases; we deal with them in turn. First, if the compound is from a reaction in a module and it stays in the module, or the compound is from the remainder and stays in the remainder, we can use minimal paths; this will preserve the module structure. If the compound is from the remainder and enters a module then recall that the net makes compounds as late as possible. So, there will necessarily be another compound which enters the other copy of the module. If that compound is produced at a later period we go on to the next compound and wait until the later period to perform the procedure detailed in the rest of this step. If both compounds are available at this period then we find the analogous input to the other copy of the module. These are either recurrents or were produced an odd number of periods earlier. If the analogous input is a recurrent, then use a minimal

path to join the compound produced at the earlier period to the module, and assign the same functional groups to the recurrent (for consistency with module structure).

Up to this point the brute-force and distance-measure methods coincide. The brute-force method proceeds to Step 6, which describes how to continue traversing the net and the table, leaving some functional groups unspecified. The method then loops sequentially over all possible assignments for these remaining functional groups, and chooses the one with the fewest functional group changes.

The distance-measure method uses an optimality criterion to finish the specification of functional groups in period 3, as far as possible. If the two compounds that enter the two modules in corresponding ways were produced from changers (i.e., neither is a recurrent) then we must find a compound to which both compounds may be converted. Denote the two compounds by cpd1 and cpd2; denote the common compound to which they will be modified by cpdmod. There will be two copies of cpdmod, one for each module. Some of the functional groups on cpdmod have already been specified by the genzyme assignment which respects the module structure.

Now, we give details for specification of functional groups that can be derived from the functional groups already assigned. To do so we use a Hamming-distance argument. Suppose a functional group is specified in both cpd1 and cpd2 and is the same for both. If the functional group is specified on cpdmod and is the same then it can be left unchanged. If the functional group is not specified on cpdmod then we specify it to be common functional group. If the functional group is specified in cpdmod differently, then we must change it in both connection paths to agree with cpdmod.

If the functional group is specified on cpd1 but not on cpd2 then it may or may not be specified on cpdmod. If it is specified on cpdmod and is the same, specify the corresponding functional group on cpd2 to be the same. If the functional group is specified on cpdmod and is different from the corresponding functional group on cpd1 then on cpd2 specify the functional group to be the same as on cpdmod. This gives a functional-group change for the connection from cpd1 to cpdmod; the connection from cpd2 to cpdmod does not change that functional group. If the functional group is specified in cpd2 but not in cpd1, the procedure is similar.

If a functional group is unspecified on cpd1 and is unspecified on cpd2, then it either is specified in cpdmod or not. If it is specified in cpdmod then use that functional group in both cpd1 and cpd2, if not then leave it unspecified.

If a functional group is specified on both cpd1 and cpd2 and they are different then either it is specified on cpdmod or not. If it is specified on cpdmod then we do not make any changes: this gives a functional-group change on the connection from cpd1 to cpdmod and a functional-group change on the connection from cpd2 to cpdmod. If the functional group is not specified on cpdmod then we use the following explicit Hamming-distance argument. (The preceding is an implicit Hamming-distance argument.)

The Hamming distance between two compounds determines the entries that are not

yet specified in cpdmod and are specified differently in cpd1 and cpd2. Normally, the Hamming distance between two vectors counts the number of entries in which the two vectors differ. Here, we use the Hamming distance on the matrix representation of two compounds and count the number of places in which they differ. More formally, we define the Hamming distance H for this case as follows. If an entry is specified on cpdmod then we ignore it and the corresponding entries in cpd1 and cpd2 which either were specified in the last period or have now been specified by the first part of Step 5. If an entry is not specified in cpdmod then the corresponding entries in cpd1 and cpd2 are necessarily specified and are different.

Consider solving the equation

$$H(\text{cpd1}, \text{cpdmod}) = H(\text{cpd2}, \text{cpdmod}), \tag{3.3}$$

for cpdmod when the number of entries not specified in cpdmod is even and solving

$$H(\text{cpd1}, \text{cpdmod}) = H(\text{cpd2}, \text{cpdmod}) \pm 1, \tag{3.4}$$

for cpdmod when the number of entries not specified in cpdmod is odd. In general, there will be several solutions; we loop over them. Solutions to equation (3.2) essentially give midpoints on line segments joining cpd1 and cpd2. The solutions are not unique because they correspond to different shortest paths in a lattice that join the two compounds. Equation (3.3) has an analogous interpretation.

A similar procedure can be used to join two modules to the remainder. That is, we have to join an output from a module to a remainder via a connection. If the other output from the other copy of the module is available, we apply the similar procedure–otherwise, we wait and do it when that compound is available.

Subject to omissions which can be readily filled in, we have now carried information to period 4 as fully as we can and so the compounds enter changers that occur then. We have not considered the possibility that an output of a module goes to a module (the same or different). This happens rarely, but can be handled by reformulating the last part of Step 5.

STEP 6: MINIMAL PATHS FOR SUBSEQUENT PERIODS. Apply Convention D to carry functional groups over to the products at period 5. Repeat Step 5 for the compounds produced so as to form connections to the next changers. Continue repeating Steps 5 and 6, looping over compounds available at subsequent periods until the whole net has been traversed. We pass through an odd number of periods in total because we begin and end with connections.

STEP 7: FUNCTIONAL GROUPS ON RECURRENTS. Once we have traversed the whole net, the only compounds that are not fully specified can arise only from recurrents. To fill in these last functional groups we proceed from right to left.

Find the earliest unspecified functional group. If it is in a module and is specified on the other copy of the module, then use that specification. Otherwise, apply Convention D, looping over the net $(l' + 1)/2$ times, where l' is the length of the longest path in the net.

Now the only functional groups which are not specified arise from recurrents. Therefore they do not change, and it does not matter what they are chosen to be. In our application, we only elaborated nets with recurrent C5s or C6s, and we choose these to be R5Ps or F6Ps respectively. In cases where the recurrent functional groups were not specified this does not matter. In cases where the functional groups should have been specified differently from those of R5P or F6P to minimize the number of steps in the net, our application might have given a few unnecessary functional group changes. However, in the detailed examination by hand we sought to eliminate any that were found.

We argue that the procedure in this appendix give a net with the fewest functional group changes among all R-nets given the genzyme assignment to the C-net, if there is a unique local minimum. First note that if there is a unique local minimum then it is also a unique global minimum. Our procedure produces a net which is minimized at each connection. This is a local minimum because any change in one functional group gives a net with more functional-group changes. By the uniqueness of the local minimum this must also be the global minimum.

REFERENCES

Atkinson, D. E. (1969). Limitation of metabolite concentrations and the conservation of solvent capacity in the living cell. *Curr. Top. Cell. Regul.* **1**, 29–63.

Baskin, A. B., R. E. Reinke and J. E. Mittenthal (1992). Exploring the role of finiteness in the emergence of structure, in *Principles of Organization in Organisms*, J. Mittenthal and A. Baskin (Eds) Redwood City, CA: Addison-Wesley, pp. 337–377.

Brown, G. C. (1991). Total cell protein concentration as an evolutionary constraint on the metabolic control distribution in cells. *J. Theor. Biol.* **153**, 195–203.

de Duve, C. (1991). *Blueprint for a Cell*, Burlington, NC: Neil Patterson.

Happel, J., P. H. Sellers and M. Otarod (1990). Mechanistic study of chemical reaction systems. *Ind. Eng. Chem. Res.* **29**, 1057–1064.

Hendrickson, J. B., D. J. Cram and G. S. Hammond (1970). *Organic Chemistry*, 3rd edn, New York: McGraw-Hill.

Jia, J., W. Huang, U. Schoerken, H. Sahm, G. A. Sprenger, Y. Lindqvist and G. Schneider (1996). Crystal structure of transaldolase B from *Escherichia coli* suggests a circular permutation of the alpha/beta barrel within the class I aldolase family. *Structure* **4**, 715–724.

Kirkwood, T. B. L., R. F. Rosenberger and D. J. Galas (1986). *Accuracy in Molecular Processes*, New York: Chapman and Hall.

Mavrovouniotis, M. L., G. Stephanopoulos and G. Stephanopoulos (1990). Computer-aided synthesis of biochemical pathways. *Biotech. Bioeng.* **36**, 1119–1132.

Meléndez-Hevia, E. (1990). The game of the pentose phosphate cycle: a mathematical approach to study the optimization in design of metabolic pathways during evolution. *Biomed. Biochim. Acta* **49**, 903–916.

Meléndez-Hevia, E. and N. V. Torres (1988). Economy of design in metabolic pathways: further remarks on the game of the pentose phosphate cycle. *J. Theor. Biol.* **132**, 97–111.

Meléndez-Hevia, E., T. G. Waddell and F. Montero (1994). Optimization of metabolism: the evolution of metabolic pathways toward simplicity through the game of the pentose phosphate cycle. *J. Theor. Biol.* **166**, 201–220.

Mittenthal, J. E., B. Clarke and M. Levinthal (1993). Designing bacteria, in *Thinking About Biology*, W. D. Stein and F. Varela (Eds), Reading, MA: Addison-Wesley.

Nuño, J. C., I. Sánchez-Valdenebro, C. Pérez-Iratxeta and E. Meléndez-Hevia (1997). Network organization of cell metabolism: monosaccharide interconversion. *Biochem. J.* **324**, 103–111.

Rawn, J. D. (1989). *Biochemistry*, Burlington, NC: Neil Patterson.

Seressiotis, A. and J. E. Bailey (1988). MPS: an artificially intelligent software system for the analysis and synthesis of metabolic pathways. *Biotechnol. Bioeng.* **31**, 587–602.

Wood, T. (1985). *The Pentose Phosphate Pathway*, Orlando, FL: Academic.

Received 8 October 1997 and accepted 29 December 1997