Gaylord, N. G., "Proposed Participation of Excited Monomer in the Free Radical-Catalyzed High Pressure, High Temperature Polymerization of Ethylene," *J. Macromol. Sci., Chemistry*, **5**, 1015 (1971).
Gear, C. W., Comm. "The Automatic Integration of Ordinary Differential Equations," *A.C.M.*, **14**, 3 (1971).
Hellwege, K. H., W. Knoppf, and W. Wetzel, "Spezifische Wärme von Polyolefinen und einigen anderen Hochpolymeren im Temperaturbereich von 30°C—180°C," *Koll. Z.*, **180**, 126 (1962).
Howell, J. A., Unpublished (1976).
Hulburt, H. M., and S. Katz, "Some Problems in Particle Technology," *Chem. Eng. Sci.*, **19**, 555 (1964).
Jackson, R. A., P. A. Small, and K. S. Whiteley, "Prediction of Molecular Weight Distributions in Branched Polymers," *J. Polymer Sci.*, **A1**, **11**, 1781 (1973).
Katz, S., and G. M. Saidel, "Moments of the Size Distribution in Radical Polymerization," *AIChE J.*, **13**, 319 (1967).
Kinkel, K., "Some Aspects of High Pressure Polymerization of Ethylene in Technical Units," Paper presented before Tenth Annual Meeting, European High Pressure Research Group, Strasbourg, (Apr., 1972).
Marano, J., "Modelling of Polymer Reactors. The Non-Isothermal Free-Radical System," Seminar presented at Northwestern University, Evanston, Ill. (Feb. 25, 1974).
Michels, A., and M. Geldermans, "Isotherms of Ethylene Up to 3,000 Atmospheres Between 0°C and 150°C," *Physica*, **9**, 967 (1942).
———, and R. S. DeGroot, "Thermodynamic Properties of Ethylene Under Pressure Up To 3,000 Atmospheres and Temperatures Between 0°C and 150°C, Tabulated as Functions of Density," *ibid.*, **12**, 105 (1946).
Mullikin, R. V., and G. A. Mortimer, "Another Look Out Long-Chain Branching," *J. Macromol. Sci.—Chem.*, **A4**, 1495 (1970).
———, "Long-Chain Branching Under Conditions of Non-Uniform Branching Probability, *ibid.*, **A-6**, 1301 (1972).
Mullikin, R. V., P. E. Parisot, and N. L. Hardwicke, "Analog Computer Studies of Polyethylene Reaction Kinetics," Paper presented at 58th AIChE Annual Meeting, Philadelphia, Pa (Dec. 5-9, 1965).
Parks, W., and R. B. Richards, "The Effect of Pressure on the Volume, Thermodynamic Properties and Crystallinity of Polythene," *Trans. Faraday Soc.*, **45**, 203 (1949).
Raff, R. A. V., and K. W. Doak, *Crystalline Olefin Polymers*, Part I., Wiley, New York (1965).
Saidel, G. M., and S. Katz, "Dynamic Analysis of Branching in Radical Polymerization, *J. Polymer Sci.*, **A-2**, **6**, 1149 (1968).
Schoenemann, K., and J. Thies, "Calculation of High Pressure Polyethylene Reactors for Polymers of Definite Structure and Properties," Tenth Annual Meeting, European High Pressure Research Group, Strasbourg (Apr., 1972); "The Calculation of High Pressure Polyethylene Reactors by Means of a Model Comprehending Molecular Weight Distribution and Branching of the Polymer," 1st International Symp. Chem Research. Eng., Carnegie Inst., Washington, D.C. (1970).
Small, P. A., Effects of Long-Chain Branching on Distribution of Degree of Polymerization," *Polymer*, **13**, 536 (1972).
———, Effects of Long-Chain Branching on Distribution of Degree of Polymerization: 2," *ibid.*, **14**, 524 (1973).
Trementozzi, Q. A., "Effect of Long-Chain Branching on Some Solution Properties of Polyethylene," *J. Polymer Sci.*, **23**, 887 (1957).
Ver Strate, G., and W. Philippoff, "Phase Separation in Flowing Polymer Solutions," *ibid., B. (Polymer Letters)* **12**, 267 (1974).
Wu, P. C., J. A. Howell, and P. Ehrlich, "Monte Carlo Simulation of Structure of Low-Density Polyethylene," *Ind. Eng. Chem. Prod. Res. Develop.*, **11**, 352 (1972).

Manuscript received September 26, 1975; revision received December 29 and accepted December 30, 1975.

Optimal Design of Pressure Relieving Piping Networks by Discrete Merging

WAI-BIU CHENG
and
RICHARD S. H. MAH

Northwestern University
Evanston, Illinois 60201

Refinery relief-header network design can be optimized by a new discrete optimization technique which requires no rounding of decision variables, no initial guesses for the solution, and no artificial termination criteria; it produces parametric solutions when the optimum is attained. The method is also faster and more compact than the alternative continuous optimization methods.

SCOPE

Pressure relieving piping networks are widely employed as a safety measure in refineries and chemical complexes. The optimal design of such a system calls for the minimization of equipment cost while maintaining conditions of maximal flows and safe pressure limits. The constrained minimization is to be attained by selecting a discrete diameter for each of the pipe sections. A typical network may contain up to several hundred such pipe sections connected to scores of process vessels and units.

The most obvious formulation of this problem [Equations (5) to (7)] limits the maximum network size to about 100 pipe sections on a CDC 6400 computer by using gradient projection methods of nonlinear programming. Murtagh (1972) proposed a dual formulation which resulted in substantial reduction of computing time and storage. However, the drawback common to all continuous optimization techniques is that the solution must be rounded to the nearest standard pipe sizes, a procedure which raises the specter of nonoptimal size selection. Moreover, the applicability of these methods depends on the specific forms of the objective functions and constraints.

Correspondence concerning this paper should be addressed to Richard S. H. Mah.

CONCLUSIONS AND SIGNIFICANCE

The proposed discrete merge method performs the discrete optimization directly. It requires neither rounding of decision variables nor initial guesses for the solution, nor termination criteria. It is applicable to a wide class of network designs. The only requirements are that the network be acyclic and that the objective functions and constraints be monotonically increasing and decreasing functions, respectively. A prototype of this method (Office of Emergency Preparedness, 1968) was, in fact, developed in conjunction with the design of offshore natural gas pipeline networks. The proposed method incorporated several improvements which greatly enhanced its capability to handle larger problems with different pressure constraints. A typical network of 79 pipe sections required only 4 CPU s and less than 16 000 words of computer storage on a CDC 6400 computer, making the optimum design of even the largest of such networks easily within the reach of a medium size computer. An additional bonus of this method is that along with the optimum design we also obtain the parametric solutions over the range covered by the design data.

Although many significant advances have been made in optimization techniques over the last two decades, their applications to process and equipment design are exceptions rather than rules in practice. A major obstacle to progress in this direction is due to the fact that design optimization must frequently contend with discrete or integer variables for which mathematical programming techniques are least well developed (Mah, 1974).

In this paper we shall discuss one significant practical application which is now amenable to systematic optimization, namely, the design of pressure relieving piping networks which interconnect various process units and vessels to the discharge zones or flares. These are commonly referred to as relief headers in refinery terminology but in fact occur widely in chemical process installation also. Figure 1 shows a typical configuration in which the root represents the flare, the terminal nodes represent the relief valves, and the branch (each labeled with an arabic numeral) represents a pipe section between two physical junctions (valves, flare, or pipe joints). The configuration of such a network is dictated by the layout of the process unit. In the subsequent discussion we shall treat both the lengths of the pipe sections and the interconnections as specified design variables.

A noteworthy feature of such a network is that it is acyclic. In terms of the commonly accepted terminology of graph theory, it is a tree. It is also convenient to employ other graph theory terminologies in our discussion. Thus we shall refer to the junctions as nodes and the number of branches incident to a node as the degree of that node. For example, the terminal nodes are of degree one. We shall also refer to the sequence of contiguous branches between each valve and the flare as a path. For instance, in Figure 1 path IV is defined by the set of pipe sections, $S_{IV} = (79, 42, 40, 38, 36, 35, 4)$. A typical relief header network may contain scores of such paths and up to several hundred branches and nodes.

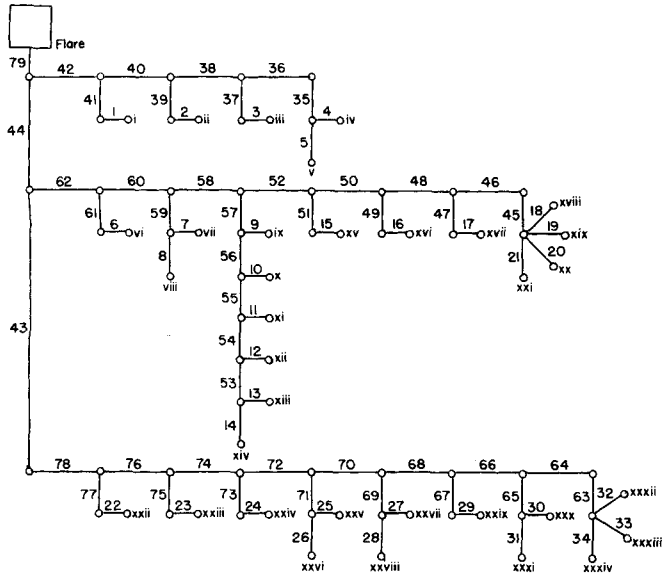

Fig. 1. A relief header network.

The design of the network calls for the selection of pipe diameters such that the discharge through each valve attains the maximum (sonic) velocity for an initial transitory period. Since the flare pressure and the process unit pressures are specified, this requirement amounts to the stipulation of a maximum allowable pressure drop over each path. Clearly, there are many feasible solutions to this design problem. The optimal design, however, will minimize the capital investment (there is no operating cost to speak of in this case) while meeting the stipulated discharge rates and pressure drop constraints.

It is common to assume that the capital investment will be proportional to the total weight of the piping network and that the weight per unit length of each pipe is a linear function of the diameter. Hence we may formulate the objective function as

$$g(\mathbf{d}) = \sum_{i=1}^{S} l_i(\alpha + \beta d_i) \qquad (1)$$

where l_i and d_i are the length and the diameter of the i^{th} pipe section and S is the total number of pipe sections. To introduce the pressure drop constraints, we need an equation which will relate the pressures to the discharge rates and the piping characteristics. In this case the customary design standard (API-RP520, 1967) stipulates the following equation:

$$P_U^2 - P_D^2 = \frac{0.3697\, fW^2RT}{\pi^2 g_c M d^5}\left(1 + \frac{4.61d}{4fl}\log_{10}\frac{P_U}{P_D}\right) \qquad (2)$$

which, strictly speaking, applies to the isothermal flow of a compressible fluid. For moderate pressure changes, the kinetic energy correction term within the bracket is close to unity (Perry and Chilton, 1973). As a first approximation, it is frequently treated as a constant, giving rise to

$$PSQ_i = P_{Ui}^2 - P_{Di}^2 = K_i/d_i^{4.814} \qquad (3)$$

and the corresponding constraint

$$b_j \geq \sum_{i \epsilon S_j} PSQ_i = \sum_{i \epsilon S_j} K_i/d_i^{4.814} \qquad (4)$$

It should be pointed out that none of the foregoing assumptions are required in the proposed method (discrete

merge). However, since they are inherent in the alternative treatment which uses nonlinear programming techniques, we have explicitly stated these assumptions in order to facilitate a direct comparison.

OPTIMIZATION WITH NONLINEAR PROGRAMMING TECHNIQUES

The constrained minimization problem stated above may be transformed into a form well suited to gradient projection methods of nonlinear programming by making the following substitution:

$$x_i = 1/d_i^{4.814} \quad i = 1, \ldots, n \tag{5}$$

whereupon we obtain a linearly constrained minimization problem

$$\min_{\mathbf{x}} \sum_{i=1}^{S} l_i (\alpha + \beta x_i^{-0.2077}) \tag{6}$$

subject to

$$\sum_{i \epsilon S_j} K_i x_i \leq b_j \tag{7}$$

$$0 \leq x_i \leq \hat{x}_i \tag{8}$$

Murtagh (1972) pointed out that rounding errors and storage limitations restrict the applicability of such techniques to networks of approximately 100 pipe sections or less. As an alternative, he proposed to solve the following dual problem:

$$\max_{\boldsymbol{\lambda} \leq 0} \min_{\mathbf{x} \epsilon E^n} L(\mathbf{x}, \boldsymbol{\lambda}) \tag{9}$$

where the Lagrangian function

$$L(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{i=1}^{S} l_i (\alpha + \beta x_i^{-0.2077}) - \sum_{j=1}^{P} \lambda_j \left(\sum_{i \epsilon S_j} K_i x_i - b_j \right) \tag{10}$$

The advantage of this approach stems from the fact that the minimization over $\mathbf{x}$ can be carried out analytically. Consequently, the dimensionality of the optimization problem is reduced from S, the number of pipe sections to P, the number of paths. Murtagh (1972) reported computer storage reduction of more than 50% and computing time reduction of up to 80% using the dual instead of primal formulation.

The success of the dual approach and the form of the objective function and constraints suggest geometric (posynomial) programming as an alternative optimization technique. In the absence of the so-called reverse constraints, the posynomial program takes the following form:

$$\min g_0(\mathbf{t}) \tag{11}$$

subject to

$$g_j(\mathbf{t}) \leq 1, \quad j = 1, 2, \ldots, P \tag{12}$$

$$\mathbf{t} \geq 0 \tag{13}$$

where

$$g_j(\mathbf{t}) = \sum_i c_i \prod_k t_k^{a_{ik}} \quad j = 0, 1, 2, \ldots, P \tag{14}$$

The coefficients c_i are positive, and the exponents a_{ik} are arbitrary real constants. In our case

$$g_0(\mathbf{d}) = \sum_{i=1}^{S} l_i (\alpha + \beta d_i) \tag{15}$$

$$g_j(\mathbf{d}) = \frac{1}{b_j} \sum_{i \epsilon S_j} K_i d_i^{-4.814}, \quad j = 1, 2, \ldots, P \tag{16}$$

We have S primal variables and $2S$ to $(P + 1)S - (P - 1)P$ terms. Using geometric programming codes available to us (Kuester and Mize, 1973; Jefferson, 1974), we have found both the storage and computing time requirements to be quite excessive. A fourteen-branch and five-path problem, for instance, took 11.9 CPU s and occupied 9 728 words of storage on a CDC 6400 computer with Blau's program used (Kuester and Mize, 1973).

In addition to the simplifying assumptions discussed earlier, a drawback common to the use of continuous optimization techniques is that the solution must be rounded to the nearest standard pipe sizes, a procedure whose pitfalls must then be eliminated by a partially enumerative search. By contrast, the proposed method of discrete merging is fast, direct, compact, and flexible.

THE DISCRETE MERGE METHOD

Given the branch lengths and network configuration, each pipe section is characterized by the pressure drop (PSQ_i), the cost (C_i), and a size index (k_i). Numerical values of these three quantities arranged in the order of increasing pipe sizes for a given pipe section form a branch list which is the basic information used in the discrete merge method. Notice that we have elected to characterize the pipe size by an integer index k rather than its diameter, since the exact physical dimension is immaterial to the algorithm. For that matter, the size increments need not even be uniform, nor do they have to be the same for different pipes. For example, $k_1 = 1, 2, 3, 4$ may correspond to $d_1 = 0.1357, 0.1626, 0.2134$, and 0.2752 m I.D., but $k_2 = 1, 2, 3$ may correspond to $d_1 = 0.1100, 0.1626, 0.2752$ m I.D. Notice also that the branch lists for different pipe sections may be of different lengths. We can thus assign a single set of entries to the branch list of a pipe section whose diameter is to be treated as fixed in a given design.

Consider a 100 branch network with branch lists of uniform length of 5. There are $5^{100} = 8 \times 10^{69}$ possible combinations of pipe diameters. Direct enumeration is, therefore, not a practical strategy even for a moderately sized network. In the discrete merge method we systematically create optimal partial assignments by merging the appropriate branch lists beginning with the branches incident to the nodes of degree one. It is a special case of dynamic programming applied to branch list processing. In creating these optimal partial assignments, we make use of the following two properties. For a given pipe length and mass flow rate, the pressure drop is a monotonically decreasing function of the pipe diameter, but the piping cost, on the other hand, is a monotonically increasing function of the pipe diameter. The merge is carried out in stages; the optimal partial assignments created in one merge become the equivalent branch lists in the next merge.

There are two kinds of merges. Parallel merge is the procedure used to assign optimal diameters to the branches (or equivalent branches) which directly connect nodes of degree one to a common node. For instance, branches 32, 33, and 34 in Figure 1, whose branch lists are shown along with that of branch 63 in Table 1. Each branch list is arranged in order of decreasing pressure drop. As a result of a parallel merge, these branch lists are merged to form an equivalent branch list which is similarly arranged in order of decreasing pressure drop, but which contains only those partial assignments that are locally

Table 1. Examples of Branch Lists

PSQ_{32}	C_{32}	k_{32}	PSQ_{33}	C_{33}	k_{33}
3 857	390	1	3 113	260	1
506	469	2	408	312	2
155	533	3	125	355	3
PSQ_{34}	C_{34}	k_{34}	PSQ_{63}	C_{63}	k_{63}
739	104	1	37 736	520	1
97	125	2	4 949	625	2
30	142	3	1 515	710	3

Table 2. An Equivalent Branch List Generated by Parallel Merging of Branches 32, 33, and 34

PSQ	C	k_e	k_{32}	k_{33}	k_{34}
3 857	754	1	1	1	1
3 113	833	2	2	1	1
739	885	3	2	2	1
506	906	4	2	2	2
408	970	5	3	2	2
155	1 013	6	3	3	2

Table 3. Serial Merging of Branch 63 (j) and the Equivalent Branch (i) in Table 2

k_j	1		2		3	
k_i	PSQ	C	PSQ	C	PSQ	C
1	41 593	1 274	8 806	1 379	5 372	1 464
2	40 849	1 353	8 062	1 458	4 628	1 543
3	38 475	1 405	5 688	1 510	2 254	1 595
4	38 242	1 426	5 455	1 531	2 021	1 616
5	38 144	1 490	5 357	1 595	1 923	1 680
6	37 891	1 533	5 104	1 638	1 670	1 723

Table 4. An Equivalent Branch List Generated by Serial Merging of Branch 63 and the Equivalent Branch in Table 2

PSQ	C	k_e
41 593	1 274	1
40 849	1 353	2
8 806	1 379	3
8 062	1 458	4
5 372	1 464	5
4 628	1 543	6
2 254	1 595	7
2 021	1 616	8
1 923	1 680	9
1 670	1 723	10

optimal. Such a list is shown in Table 2. The key to the parallel merge algorithm is that only the optimal partial assignments are generated in the process.

To simplify the presentation, let us suppose that the upper bound of pressure drops for each path is the same; that is, b_j for all paths are equal. Then, for any given assignment of diameters, the pressure drop of the equivalent branch will be determined by the branch with the largest PSQ. Increasing the size of other branches will only increase the cost without the benefit of lowering the limiting PSQ. Hence, starting with the combination of largest branch PSQ's (the left most entries of the branch lists in Table 1), only the assignments generated by successively increasing the sizes of the limiting branches will be locally optimal. With reference to Table 2, the first partial assignment is limited by branch 32 and the second by branch 33, and so on.

We shall now outline the steps in the parallel merge of n branches:

P1. Set $k_i = 1$ for $i = 1, 2, \ldots, n$ and $k_e = 1$

P2. Let $\hat{PSQ} = \max_i \{PSQ_{i,k_i}\}$ and $r = \{i | PSQ_{i,k_i} = \hat{PSQ}\}$.

P3. Form the k_e^{th} entries of the equivalent branch list:

$$PSQ_{k_e} = \hat{PSQ} \text{ and } C_{k_e} = \sum_{i=1}^{n} C_{i,k_i}.$$

P4. Stop, if k_r is the last entry of branch list r.

P5. Otherwise, $k_r := k_r + 1$ and $k_e := k_e + 1$ and go to step P2.

In the illustration, the merged equivalent list contains 6 of a total of 27 possible partial assignments.

Serial merge is used to create optimal partial assignments for two branches linked through a common node of degree two, if at least one of the two branches is also incident to a node of degree one or is an equivalent branch. For example, branch 63 and the equivalent branch formed from the parallel merge of branches 32, 33, and 34 are candidates for a serial merge. Table 3 lists the 3×6 possible combinations of diameter assignments in such a serial merge. To create an equivalent branch list, the entries in this table must be sorted in descending order of PSQ. However, it may be noted that each row and each column of this table are already in descending order of PSQ. Hence we need to carry no more than h entries at a time, where h is the shorter of the two branch list lengths. In this case $h = 3$.

It should be pointed out that whereas we start with branch lists whose entries are arranged in ascending order of costs, the merged entries may contain entries whose costs are higher than their successors. Since such entries clearly cannot possibly be optimal partial assignments, they may be eliminated from the merged equivalent branch list.

Let i and j be the two branches, let e be the equivalent branch after the serial merge, and let $h_j \leq h_i$. We shall define

$$PS_{k_i,k_j} = PSQ_{i,k_i} + PSQ_{j,k_j} \tag{17}$$

$$CS_{k_i,k_j} = C_{i,k_i} + C_{j,k_j} \tag{18}$$

Since in the process of serial merge k_i may be different for different k_j, we shall also use $k_i(k_j)$ to indicate the k_i associated with a particuler k_j. The steps in a serial merge are as follows:

S1. Set $k_e = 1$ and $k_i = 1$ and the index set, $I = \{1, 2, \ldots, h_j\}$.

S2. Let $\hat{PS} = max \{PS_{k_i(k_j),k_j} | k_j \epsilon I\}$ and let $r = \{k_j | PS_{k_i(k_j),k_j} = \hat{PS}\}$.

S3. If $k_e \neq 1$ and $C_{k_e-1} \geq CS_{k_i(r),r}$, $k_e := k_e - 1$ and repeat the step.

S4. Form the k_e^{th} entry of the equivalent branch list:

$PSQ_{k_e} = \hat{PS}$ and $C_{k_e} = CS_{k_i(r),r}$.

S5. If $k_i(r) = h_i$, delete r from I. Otherwise let $k_i(r) := k_i(r) + 1$ and go to step S2.

S6. If $I = \phi$, stop. Otherwise, go to step S2.

Table 4 shows the equivalent branch list formed from the serial merge of branch 63 with the equivalent branch 32/33/34.

We shall now turn to the more general case for which the b_j's may be different. Let one of the paths, say r, be the reference path, and let the PSQ in the branch list of each of the terminal branches i be modified as

$$PSQ_i := PSQ_i - b_j + b_r, \quad i \epsilon S_j \tag{19}$$

Note that this modification will not affect the outcome of a serial merge, since all the entries in a branch list will be altered by the same amount. For a parallel merge, on the other hand, both the absolute values of PSQ and the relative differences for the different branches have changed by amounts equal to $(b_j - b_r)$. Since each path contains only one terminal branch, the modification on a terminal branch list is equivalent to the same change on the corresponding path. By modifying the terminal branch lists in this manner, parallel and serial merges can also be applied to the general case with unequal b_j's.

Through repeated applications of parallel and serial merges, the discrete merge method eventually transforms a tree network to a single equivalent branch list which gives the optimal diameter assignment and minimal cost not only for the desired b_j's but also for all the parametric cases of different b_r values which are covered by the information inherent in the branch lists. This parametric capability is particularly useful when we analyze the trade offs between pressure drop constraints and the network cost.

COMPUTATIONAL ENHANCEMENTS

The algorithm described above works extremely well with small networks. But for large networks, the equivalent branch lists may grow so long that the computing time becomes excessive. Before we discuss the various techniques of enhancing its performance, an analysis of the characteristics of the algorithm is in order.

For the serial merge of two branch lists of lengths h_i and h_j, respectively, the number of operations is proportional to $h_i \times h_j$. The length of the equivalent branch list h_s is bounded by

$$h_i + h_j - 1 \leq h_s \leq h_i \times h_j \tag{20}$$

For a parallel merge, on the other hand, the number of operations is directly proportional to the length of the equivalent branch list h_p, since only optimal partial assignments are generated and examined. The minimal length is obtained if the limiting branch is the branch with the shortest branch list. By contrast, the maximum length is obtained if all but one of the k_i's are increased to just below their maximal allowable sizes. The implication is that the limiting branch rotates from assignment to assignment among the n branches. Hence

$$\min_i h_i \leq h_p \leq \sum_{i=1}^{n} h_i - 2(n-1) \tag{21}$$

We can also estimate the number of serial merges required in processing a given network. Consider the following procedure of reconstructing the given network. Starting with the root of the tree, the branches are added one at a time. Each time a branch is added, either the number of serial merges is increased by one or the number of paths is increased by one. Hence, for a network of S branches, the number of serial merges N_s is given by

$$N_s = S - P \tag{22}$$

For parallel merges, the estimate is less exact. If the network is a binary tree, the number of parallel merges is

$$N_p = S/2 \tag{23}$$

More generally, for a network with an average degree of nodes of m

$$N_p \leq S/m \tag{24}$$

Relations (20) and (21) show that whereas h_p is proportional to h, the dependence of h_s on h is more than linear. Since P is usually less than $S/2$, $N_s > N_p$. Moreover, the computing time for a serial merge is proportional to h^2. The analysis suggests that careful control of the growth of list lengths will do much to reduce the computing time for serial merges, which will in turn reduce the overall computing time requirement.

We shall now outline the three methods for controlling the list lengths which have been implemented and evaluated in our computer program. The first and most effective method is the use of a more coarse grid (fewer discrete pipe sizes) with the grid modified from iteration to iteration. For instance, if the original branch lists contain nine pipe sizes ($k_i = 1, 2, \ldots, 9$), only four sizes ($k_i = 1, 3, 6$, and 9) are used initially. Suppose that the first iteration assigns size 6 to pipe i. Then the grid is modified to $k_i = 4, 5, 6, 7$ for the second iteration. If the optimal assignment for pipe i is now size 5, the grid is modified to $k_i = 3, 4, 5, 6$ for the third iteration, and so on. Notice that the grid has been chosen deliberately to bias towards the lower size range because the reassignment tends to give smaller k on the average. In this example we choose to use only four sizes, which, in our experience, has proven to be satisfactory. The iteration should continue as long as the assigned size of any one branch coincides with its upper or lower size in the coarse grid, since the reassignment of one branch can affect the optimal assignment for the network as a whole. In practice, we have found three iterations to be generally adequate.

This modification is extremely effective for serial merges in a deep network. For the example shown in Figure 1, serial merges of up to thirteen branch lists are involved. The use of four sizes instead of nine reduces the computing time for the serial merges by a factor of $9^{13}/(4^{13} \times 3) = 12\,626$.

The second modification is to truncate the equivalent branch lists by deleting entries whose pressure drops exceed the specified values, that is, introducing an addition step after step S2 in serial merges:

S2a. If $\hat{PS} > b_r$, go to step S5.

If judiciously employed, this modification need not materially affect the parametric capability.

The third modification is to limit the length of an equivalent branch list to an empirically determined maximum value. Just as the second modification is aimed at truncating the list at the high PSQ end, this stratagem checks the growth of low PSQ entries on the list. It could be deployed in conjunction with the use of more coarse grids. For instance, if we limit the branch lists to four sizes, we have found empirically that a maximum length of 50 is a safe limit for equivalent branch lists. The hazard of this modification is that it may result in nonoptimal assignments for entries with low PSQ in the equivalent branch list.

PERFORMANCE EVALUATION

The discrete merging method with enhancements has been programmed and evaluated on a CDC 6400 computer. It was applied to each of the three sections as well as to the entire network shown in Figure 1. The computing time and storage requirement for each problem are summarized in Table 5. Over the ranges tested,

Table 5. Computing Time and Storage Requirements for the Sample Problems

Number of branches S	Number of paths P	Computing time (CPU s)	Storage requirements (words)
14	5	0.759	11 456
30	13	1.693	12 160
35	16	1.733	12 544
79	34	3.998	15 680

we have found that both increase linearly with the number of branches. Since the configurations of most refinery relief headers are not too dissimilar to Figure 1, moderate extrapolation of performance may be made for comparison purpose. On this basis we would estimate that for medium to large networks, the discrete merge method is about two to four times faster than the best previously known procedure (Murtagh, 1972) and requires only about one third as much storage.

DISCUSSIONS

The concept of parallel and serial merges was originally developed in conjunction with the design of offshore natural gas pipeline networks (Office of Emergency Preparedness, 1968) which generally contain no more than 50 nodes. In this paper, we have enhanced the basic approach and made it applicable to a wide class of network design problems, of which the pressure relieving and natural gas gathering networks are but two special cases. The basic requirements are that the network be acyclic and that the objective functions and constraints be monotonic. No other conditions are placed on either the form of the cost function or the form of the constraints. The approach is thus far less restrictive than the alternative techniques based on nonlinear programming.

For pipeline network design for which the basic size variables are discrete, the discrete merge method is direct, requiring neither rounding of decision variables, initial guesses for the solution, nor accuracy criteria for the termination of the procedure. It is also very fast and compact. As a final bonus, parametric solutions are provided when the optimum design is attained.

ACKNOWLEDGMENT

Acknowledgment is made to the Donors of the Petroleum Research Fund, administered by the American Chemical Society for the support of this research, and to Dr. Bruce A. Murtagh for his suggestion of the test problems.

LITERATURE CITED

API-RP520, *Recommended Practice for the Design and Installation of Pressure-Relieving Systems in Refineries: Part 1 —Design*, 3 ed., American Petroleum Institute (1967).

"Design of Economical Offshore Natural Gas Pipeline Systems," *Report R-1*, Office of Emergency Preparedness (1968).

Jefferson, T. R., "Manual for the Geometric Programming Computer Code [GPROG(CDC) Version 2]" *Rept. No. 1974/OR/2*, The University of New South Wales (1974).

Kuester, J. L., and J. H. Mize, *Optimization Techniques with FORTRAN*, pp. 135-154, McGraw-Hill, New York (1973).

Mah, R. S. H., "Recent Developments in Process Design," in *Basic Questions of Design Theory*, William R. Spillers, ed., North-Holland (1974).

Murtagh, B. A., "An Approach to the Optimal Design of Networks," *Chem. Eng. Sci.*, 27, 1131 (1972).

Perry, J. H., and C. H. Chilton, ed., *Chemical Engineer's Handbook*, 5 ed., pp. 5-26, McGraw Hill, New York (1973).

NOTATION

a = exponents in Equation (14)
b_j = upper bound of pressure drop constraint for path j in Equation (4)
c = coefficients in Equation (14)
C = branch cost
CS = sum of C's, defined by Equation (18)
d = branch diameter, m
E^n = Euclidean n space
f = friction factor, $0.0475\ (2.8186 \times 10^6\ W/\mu d)^{-0.186}$
g, g_o = objective function or constraint in design optimization
g_c = gravitational constant, $kg\text{-}m/(N - s^2)$
h = length of branch list
k = size index in the discrete merge method and free subscript in Equation (14)
K = coefficient in Equation (3)
l = length of pipe section, m
m = average degree of nodes
M = molecular weight, kg/kg mole
n = number of branch lists undergoing a parallel merge
N = number of merges
P = number of paths
P_U = pressure at upstream node of branch, N/m^2
P_D = pressure at downstream node of branch, N/m^2
PS = sum of PSQ's, defined by Equation (17)
PSQ = pressure drop function, defined by Equation (3)
R = gas constant, 8314.4 $N\text{-}m/(kg\ mole\text{-}^\circ K)$
S = number of branches
S_j = set of branches in path j
t = independent variable in Equations (11) to (14)
T = temperature, °K
W = mass flow rate, kg/s
X = $1/d^{4.814}$, the transformed decision variable
α, β = coefficients relating the weight of a pipe of unit length to its diameter, in Equation (1)
λ = Lagrangian multiplier, in Equation (10)
μ = viscosity, $N\text{-}s/m^2$
ϕ = null set

Subscripts

e = equivalent branch
i = branch index or independent variable index
j = path index or constraint index
k = size index
p = parallel merge
r = reference branch or as defined in step P2 of the parallel merge algorithm
s = serial merge
u = dummy subscript used in the algorithm of parallel merge

Superscripts

$\wedge$ = maximum value

Other Symbols

$:=$ = replaced by
ϵ = belongs to

Manuscript received November 20, 1975; revision received January 6, and accepted January 7, 1976.