ELSEVIER

Advances in Engineering Software 32 (2001) 239–253

ADVANCES IN ENGINEERING SOFTWARE

www.elsevier.com/locate/advengsoft

A heuristic algorithm for drawing of a flow diagram

D.N. Grekas, C.A. Frangopoulos*

Department of Naval Architecture and Marine Engineering, National Technical University of Athens, Heroon Polytechniou 9, 157 73 Zografou, Greece

Received 18 December 1998; accepted 27 May 2000

Abstract

A flow diagram is a graphical presentation of an energy or chemical system with its components and their interconnections through mass and energy streams. An automatic drawing algorithm of flow diagrams has been developed and presented in this article. It heuristically imitates all the procedures carried out by a designer, starting with his conceptual understanding of the system's topological structure and finishing with graphically representing the system on the paper or screen. The topological structure of the system is given as input in the form of digraph. As a first step, it is transformed to a planar digraph by introducing new vertices representing crossings between the various streams. The near-optimum (smallest) number of crossings is determined heuristically. Then the flow diagram is drawn on the screen using special mechanical engineering symbols for the components of the system. Horizontal and vertical lines represent streams of mass or energy, identified by different colors. Unnecessary expansion of the drawing area is avoided by the application of linear and integer-linear programming algorithms. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Intelligent CAD; Heuristics; Graph-models; Flow diagram; Engineering systems

1. Introduction

One significant task during the design of a system is to create a picture that shows the structure of the system, i.e. the components and their interconnections. In the case of a thermal system (power plant, cogeneration system, refrigeration unit, etc.), this picture is given by the flow diagram. The designer's conceptual work is drawn on paper or on screen, either manually or with the aid of computer aided design software (CAD). This process is tedious and time consuming, more so if several alternative solutions (diagrams) have to be drawn for evaluation, comparison and final selection.

It would be of great help to the designer if his conceptual work could be automatically transformed to a diagram with the aid of an appropriate software tool. One approach to encounter the problem of automatic design is through expert systems. A good example of a ruled-based expert system in architectural design is found in Ref. [10]. Our approach comes under the field of intelligent CAD. An algorithm for such a purpose has been developed and presented in the following. It makes use of the theory of planar digraphs [1,2,4,5]. A *plane digraph* is a digraph with no crossings between edges. A digraph, which can be transformed to a plane digraph, is called *planar digraph*. The algorithm has been developed in C++ using various data structure algorithms, including data structures for digraph representation [6–9].

The input to the algorithm is a digraph representing the topological structure of the system. The algorithm is heuristic and it implements the steps that a designer follows, in order to draw not only a correct but also an aesthetically acceptable diagram. Speaking rigorously, a *heuristic algorithm* is a decision tree searching method, which takes advantage of expert's information about searching priorities [3,14]. In the algorithm, the search stops when the first correct and aesthetically acceptable diagram is found. It doesn't attempt to find a better one by searching further. In this way, the execution time is significantly reduced. However, the algorithm may be easily extended to a truly heuristic algorithm, if an appropriate *heuristic function* is defined, which determines a numerical value for the produced diagram quantifying the aesthetic result, even though this may be subjective.

Two of the main characteristics of the algorithm are the following: (i) it heuristically finds the near optimum (smallest) number of crossings between streams, and (ii) it avoids the unnecessary expansion of the drawing area. The algorithm is based on the appropriate transformation of the problem to: an integer linear programming (ILP) problem, which is solved by the Branch and Bound method [12,13];

* Corresponding author. Tel.: +30-1-772-1108; fax: +30-1-772-1117.
E-mail address: caf@naval.ntua.gr (C.A. Frangopoulos).

0965-9978/01/$ - see front matter © 2001 Elsevier Science Ltd. All rights reserved.
PII: S0965-9978(00)00086-7

Nomenclature

$d_{i,j}$	direction number of the edge connecting vertices V_i and V_j
$\bar{d}_{i,j}$	direction number of same direction edges starting with V_i and ending with V_j
$e_{i,j}$	number of edges belonging to boundary hyperedge
E	the number of boundary hyperedges of a digraph
E_k	alternative symbol for edge of a digraph (by its identity number)
$i–j$	set of same direction edges starting with V_i and ending with V_j
$l_{i,j}$	length of edge (V_i, V_j)
R	the number of regions of a plane digraph (including the external region)
R_i	region of a digraph OR vertex of its conjugate digraph
(R_i, R_j)	boundary hyperedge, i.e. the common boundary between regions R_i and R_j
V	the number of vertices of a digraph
V_i	vertex of a digraph
(V_i, V_j)	edge of a digraph connecting vertices V_i and V_j
Greek symbols	
$\lambda_{i,j}$	angle between the first and the last edge of the boundary hyperedge (R_i, R_j)
$\lambda^{+}_{i,j,k}$	$\lambda_{i,j}$ or $\lambda_{j,i}$ depending on the relative position of (R_i, R_j) with respect to its end vertex V_k
$\lambda^{-}_{i,j,k}$	$\lambda_{i,j}$ or $\lambda_{j,i}$ different than $\lambda^{+}_{i,j,k}$
ϕ_i	size of an angle in number of right angles, if its vertex has degree ≥ 2; $\phi_i = 1, 2$
ω_i	size of an angle in number of right angles, if its vertex has degree $= 2$; $\omega_i = 1, 2, 3$

and to a linear programming (LP) problem, which is solved by the Simplex method [11–13].

The following analysis refers to flow diagrams of thermal systems. However, it can be easily extended to diagrams of other engineering systems such as electrical, electronics, water piping, etc.

2. The main parts of the algorithm

A key idea of the algorithm is to imitate all the procedures carried out by a good designer. For this purpose, the procedure followed by a designer is divided into steps, which then are translated into mathematical and computer science language. The first level of this division consists of three subproblems, which the designer should solve sequentially. These are the following:

Problem 1 (modification of the topology). Given the topological structure of the system as a digraph, transform it to a digraph with an equivalent flow information with each vertex having degree ≤4. Check if this digraph is planar. If it is not, force it to be planar by heuristically introducing a near-optimum (smallest) number of new vertices representing crossings between edges. Transform the planar digraph to a plane digraph and identify the regions, the angles and the new introduced edges/vertices of the final plane digraph.

A well-known “planarity testing algorithm” [1] (it tests if a graph is planar or not) is extended here, in order not to stop if it detects that the digraph is not planar, but to continue and gradually transform it to planar by introducing new crossing vertices.

Problem 2 (identification of the geometry). Given the output plane digraph of Problem 1, with crossings, properly assign integer values 1, 2 or 3 to angles. These values express its size in multiples of the right angle, in such a way that it is possible for the produced digraph to fit on a hidden, uniform, orthonormal square grid. In order for the procedure above to be feasible, probably some edges will be recursively divided in two successive edges with a new vertex between them and two new angles will be produced. The assignment of the integer values to all angles and the decisions about which edges will recursively be divided must lead to a new digraph with the minimum possible number of edges (or, equivalently, the minimum number of edge divisions).

Then, the length of all the edges is adjusted in order to fit the digraph on a hidden, uniform, orthonormal square grid to avoid unnecessary expansion of the drawing area. This procedure will assign integer numbers to edges, which will express their lengths as multiples of the length of the edge of the unit square of the hidden, uniform, orthonormal square grid. Also, in a similar way, integer coordinates are assigned to vertices.

Problem 3 (engineering drawing). The correspondence between the components of the input digraph (vertices and edges) and the components of the thermal system is given. The rules for drawing the flow diagram are specified and implemented. Using the aforementioned and the results of Problem 2, the flow diagram is drawn on the screen.

3. The data structure

Before describing the algorithms corresponding to the three problems, the underlying data structure that is based on a digraph is presented. A digraph is a data structure

دریافت فوری ← متن کامل مقاله

ISIArticles
مرجع مقالات تخصصی ایران

- ✓ امکان دانلود نسخه تمام متن مقالات انگلیسی
- ✓ امکان دانلود نسخه ترجمه شده مقالات
- ✓ پذیرش سفارش ترجمه تخصصی
- ✓ امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
- ✓ امکان دانلود رایگان ۲ صفحه اول هر مقاله
- ✓ امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
- ✓ دانلود فوری مقاله پس از پرداخت آنلاین
- ✓ پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات