ON THE COMPLEXITY OF GENERAL GRAPH FACTOR PROBLEMS

by

D.G. Kirkpatrick and P. Hell

Technical Report 81-7

August 1981

ON THE COMPLEXITY OF GENERAL GRAPH FACTOR PROBLEMS

by

D.G. Kirkpatrick
Department of Computer Science
University of British Columbia

P. Hell
Departments of Computing Science and Mathematics
Simon Fraser University

Abstract

For arbitrary graphs G and H, a G-factor of H is a spanning subgraph of H composed of disjoint copies of G. G-factors are natural generalizations of 1-factors (or perfect matchings), in which G replaces the complete graph on two vertices. Our results show that the perfect matching problem is essentially the only instance of the G-factor problem that is likely to admit a polynomial time bounded solution. Specifically, if G has any component with three or more vertices then the existence question for G-factors is NP-complete. (In all other cases the question can be resolved in polynomial time.)

The notion of a G-factor is further generalized by replacing G by an arbitrary family of graphs. This generalization forms the foundation for an extension of the traditional theory of matching. This theory, whose details will be developed elsewhere, includes, in addition to further NP-completeness results, new polynomial algorithms and simple duality results. Some indication of the nature and scope of this theory are presented here.

Key words: algorithms, complexity, factor, graph, matching, NP-completeness.

1. Introduction.

Let H denote an arbitrary graph with vertex set V(H) and edge set E(H). A matching in H is any subset M of E(H) such that no two elements of M have a vertex in common. A matching M is perfect (also called a 1-factor) if exactly one element of M is incident with each vertex in V(H). If H is a weighted graph then the weight of a matching M is just the sum $\sum_{e \in M} \text{weight}(e)$.

The notion of a matching in a graph has numerous applications in such diverse areas as transversal theory, assignment problems, network flows, multiprocessor scheduling, shortest path algorithms, and the Chinese Postman and Traveling Salesman problems [3, 10, 12, 13, 16, 17, 22, 31, 33, 36]. The existence of polynomial time bounded algorithms for the construction of matchings of maximum cardinality (and hence determing the existence of perfect matchings) or maximum weight is well known [6, 7, 8, 9], although the exact complexity of the problems is not yet settled and work continues on this aspect of the problem [23, 26]. In addition there is a rich mathematical theory that has developed about the matching problem that includes characterizations of graphs that admit perfect matchings [39] and (more generally) duality theorems on maximum matchings [1, 8, 12, 17, 32, 33, 34].

A matching in H may be viewed as a collection of disjoint subgroups of H , each isomorphic to K_2 .† In a perfect matching the

---

† K_t denotes the complete graph on t vertices.

vertex set $V(H)$ is completely partitioned by the vertex sets of the subgraphs. This suggests the following natural generalization:

Let G be an arbitrary graph. A G-packing of a graph H is a set $\{G_1, \ldots, G_d\}$ of disjoint subgraphs of H such that each G_i is isomorphic to G. (Note that we do not require the G_i's to be induced subgraphs; that variant of the problem is discussed later in the paper). A perfect G-packing or G-factor of a graph H is a G-packing such that the sets $V(G_i)$ partition $V(H)$. Clearly, a K_2-packing is just a matching and a K_2-factor is a perfect matching.

We were motivated to study this generalization of matching by both practical and theoretical considerations. Graph partitioning problems arise in a number of applications [2, 4, 11, 15, 21, 24, 29]. Our original motivation [19, 30] stemmed from the study of examination scheduling. After an assignment of courses to examination periods, eliminating what could be called first-order conflicts (essentially a graph colouring problem), has been accomplished, the problem arises of assigning the examination periods to real time periods, under fairly standard constraints (normally some k examination periods are scheduled in sequence each day). The objective here is to minimize second-order conflicts (or inconveniences). Typically, this might include an occurrence of a student writing two examinations on the same day, or perhaps two consecutive examinations on the same day. Suppose H is the graph whose vertex set is the set of examination periods $\{p_i \mid 1 \le i \le t\}$ and whose edge (p_i, p_j) is weighted by the number of students common to courses examined in periods p_i and p_j. Then minimization of the types of second-order conflicts illustrated above corresponds to the construction of minimum weight K_3- and P_3† - factors in H .

As we shall see, our investigation of generalized matching can also be viewed as furthering our understanding of the perceived threshold between NP-complete and polynomial-time-solvable problems [5, 14, 27, 28]. Specifically, it is of interest to know which members of this family of problems admit polynomial-time-bounded algorithms and which, like the general subgraph isomorphism problem of which they are all special instances, are NP-complete. We are able to provide a complete characterization (in the above sense) of the complexity of finding a G-factor. This characterization is similar, in spirit, to the results of Schaefer [38], Yanakakis [40] and Lewis [35] each of which establishes NP-completeness results over a broad family of interesting problems.

While our results here are essentially negative, it should be noted that an extension of the notion of G-packings and G-factors (replacing G by a family of graphs) has pointed the way to a very natural setting in which to extent the traditional theory of matching, giving rise to new polynomial algorithms and simple duality results [19, 20].

---

† P_t denotes the path on t vertices.

2. Generalizations of Matching.

Our notion of a G-packing (and G-factor) is by no means the only natural extension of the familiar concept of matching. Indeed, in section 5, we introduce and motivate the notion of a $\mathcal{G}$-packing (and $\mathcal{G}$-factor) where $\mathcal{G}$ denotes a family of graphs. This extension subsumes, and should not be confused with, the notion of F-factor introduced by Muhlbacher [37].

If the concept of matching is extended in the natural way to hypergraphs the problem of determining the existence of a perfect matching is known to be NP-complete. Karp [27] describes what is probably the simplest version of this problem as three-dimensional matching:

INSTANCE: An integer p and a set $U \subseteq \{1,2,\ldots,p\}^3$.

QUESTION: Is there a subset $W \subseteq U$ of cardinality p such that no two elements of W agree in any coordinate?

It should be clear that the k-dimensional matching problem (replace three by k above) is also NP-complete, when $k \geq 3$. The two-dimensional matching problem is equivalent to the matching problem for bipartite graphs.

Expressed as a language recognition problem, the existence problem for G-factors, which we denote FACT(G), becomes:

INSTANCE: A graph H .

QUESTION: Does H admit a G-factor?

The problem $\mathrm{FACT}(K_1)$ is trivial since every graph admits a K_1-factor. Furthermore, $\mathrm{FACT}(K_2)$ is just the question of existence of a perfect matching, and hence $FACT(K_2) \in P$. More generally, if $G = \alpha \cdot K_1 \cup \beta \cdot K_2$, that is the disjoint union of α copies of K_1 and β copies of K_2 (or, equivalently, if each connected component of G has at most two vertices), then H admits a G-factor if and only if $|V(H)|$ is divisible by $|V(G)|$ and H admits a matching with at least $\beta \frac{|V(G)|}{|V(G)|}$ edges. Thus then the usual algorithms for finding a maximum matching (eg. [7, 34]) may be used to answer FACT(G) in polynomial time. Our central result suggests that all other problems FACT(G) are unlikely to admit efficient solutions.

Theorem 4.2. If G is not of the form $\alpha \cdot K_1 \cup \beta \cdot K_2$ then FACT(G) is NP-complete.
Two important instances of this result, $G = K_3$ and $G = P_3$, were established earlier by T. Schaeffer [14,28] and D.S. Johnson [25].

The proof of Theorem 4.2 is deferred to Section 4. The following lemma allows us to restrict our attention to problems FACT(G) where G is a connected graph.

Lemma 2.1. Let G be a graph and G' any component of G with the maximum number of edges. Then, $FACT(G') \leq_P FACT(G)$.

Proof. Suppose G' has p vertices and suppose G has r distinct components isomorphic to G'. If H is any graph with dp vertices, then let T(H) denote the graph $H \cup d(G - G')$. Obviously, if H admits a G'-factor then T(H) admits a G-factor. Suppose T(H) admits a G-factor F. F must contain exactly dr components isomorphic to G'. But, by the maximality of G', the restriction of F to $d(G-G')$ contains at most $d(r-1)$ components isomorphic to G'. Hence the restriction of F to H must be a G'-factor of H. Thus H admits a G'-factor if and only if $T(H)$ admits a G-factor. □

3. Basic Modules and their Properties.

Our objective in this and the next section is to demonstrate how, for an arbitrary connected graph G on k vertices, the k-dimensional matching problem can be polynomially reduced to the problem FACT(G). Our construction is component-based (cf. [14]) in nature; in this section we describe the components (which we call modules) and their properties that we exploit in the general construction.

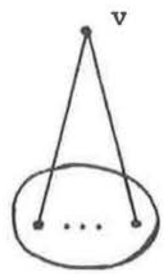
3.1. Modules and Coherences.

A module is a graph M with non-empty subset $C \subseteq V(M)$ of distinguished vertices. We call the elements of C (respectively V(M)-C) connector vertices (respectively interior vertices) of M . A G-module is any module that admits a G-packing covering all of its interior vertices (plus some, possibly empty, subset of its connector vertices).

A modular extension of the module M is any graph H , containing M as an induced subgraph, in which no interior vertex of M is adjacent to a vertex of H-M (that is, M is connected to the rest of H only through its connector vertices). Let $\pi = \{G_1, \ldots, G_d\}$ be any G-packing of some modular extension H of M . A vertex v of M is said to be bound to M by π , if $v \in V(G_i)$ implies $V(G_i) \subseteq V(M)$. A G-module M is internally G-coherent if every G-factor of every modular extension of M binds to M all of its interior vertices (that is, it respects the modularity of M).

The simplest example of an internally G-coherent G-module is the (connected) graph G itself with any one vertex $v \in V(G)$ designated as a connector vertex. We depict this schematically as:

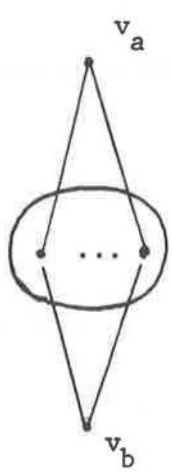
G :



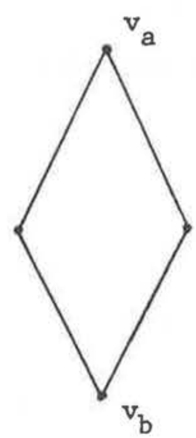
3.2. Diamond Modules.

If G is any connected graph and $v \in V(G)$, then the graph, formed from G by splitting v into two non-adjacent vertices v_a and v_b, each of which is adjacent to all of the neighbours of v in G, is called a G-diamond, and is denoted D[G;v]. We depict D[G;v] schematically as

D[G;v]:



or simply



If v_a and v_b are taken as connector vertices then D[G;v] is a G-module. Its coherence, it turns out, depends on the choice of vertex v, but a choice ensuring G-coherence always exists. Specifically, let v^* be any vertex of G that is not a cutpoint and belongs to a biconnected component of G containing at most one cutpoint. Every graph G is a guaranteed to contain at least one such vertex (cf. [18], p. 36].

<u>Lemma 3.1.</u> The module D[G;v*] with v_a^* and v_b^* as connectors is internally G-coherent.

<u>Proof.</u> Let H be any modular extension of D[G;v*] and let φ be any G-factor of H . φ induces a partition π of the interior vertices of D[G;v*]. Since D[G;v*] has exactly two connector vertices and each graph in φ is connected, π has at most two cells. All of the vertices of D[G;v*] that do not belong to the same biconnected component as v_a^* must belong to the same partition of π (otherwise there must be two vertex-disjoint paths from this set to v* in G , contradicting the choice of v*). Hence, if π has two cells then some element of φ must contain all of the vertices of D[G;v*] that do not belong to the same biconnected component as v_a^* , either v_a^* or v_b^* , and at least one vertex of H-D(G;v*]. But this component has at least one more cutpoint (namely v_a^* or v_b^*) than G , a contradiction. Thus, π has exactly one cell and hence D[G;v*] is internally G-coherent. □

We can summarize the relevant properties of diamond modules as follows:

<u>Property 3.2.</u> (a) Every G-factor of every modular extension of D[G;v*] binds to D[G;v*] its interior vertices plus exactly one of its connector vertices.

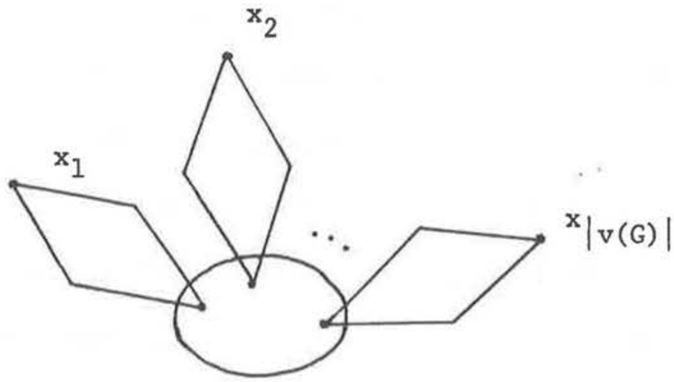
(b) The graph D[G;v*] minus either one of its connector vertices admits a G-factor.

Thus, diamond modules, wherever they appear in a larger graph, force a "choice" of one or the other of their connector vertices.

3.3. <u>Star Modules</u>.

A <u>G-star</u>, denoted S[G;v], is the graph formed from G by identifying, with each vertex w ∈ V(G), the v_a-connector of a distinct copy of D[G;v]. If the v_b-connectors of the $|V(G)|$ G-diamonds used in the construction, relabelled as $x_1,\ldots,x_{|V(G)|}$, are taken as connector vertices, then S[G;v] is a G-module. We depict S[G;v] schematically as:

S[G;v]:



S[G;v] can be seen as a modular extension of $|V(G)|$ disjoint copies of D[G;v]. As would be expected the coherence of S[G;v] depends on the coherence of D[G;v]. Specifically,

Lemma 3.3. If D[G;v] is internally G-coherent then so is S[G;v].

Proof. Let H be any modular extension of S[G;v] and let φ be any G-factor of H . H must also be modular extension of each of the $|V(G)|$ copies of D[G;v] used in the construction of S[G;v]. Since D[G;v] is internally G-coherent, it follows that all of the interior vertices of S[G;v] that are internal to one of the copies of D[G;v] must be bound to S[G;v] by φ . Since none of the remaining interior vertices of S[G;v] are adjacent to any of the connectors of S[G;v], it follows that φ binds to S[G,v] all of its interior vertices. □

Corollary 3.4. S[G;v*] is internally G-coherent.

In fact, S[G;v*] satisfies the following somewhat stronger property. A G-module M is <u>G-coherent</u> if it is internally G-coherent, if every G-factor of every modular extension of M binds to M either all or none of its connector vertices, and if in addition, both M and $M-C$ admit G-factors. G-coherence places a strong restriction on the G-modularity of M. It is clear from the definitions that G-coherent modules M, wherever they appear in a larger graph, can be viewed as forcing a "choice" of either all or none of their connector vertices (both of which are possible). Our central construction rests on the following:

<u>Lemma 3.5.</u> $S[G,v^*]$ is G-coherent.

<u>Proof.</u> Let φ be any G-factor of any modular extension of $S[G,v^*]$. Note that $S[G,v^*]$ contains $|V(G)|^2$ vertices of which $|V(G)|$ are connectors. Since $S[G,v^*]$ is internally G-coherent and the total number of vertices bound to $S[G,v^*]$ by φ must be a multiple of $|V(G)|$, it follows that either all or none of the connector vertices must be bound to $S[G,v^*]$ by φ. □

4. The General Construction.

We are now prepared to state and prove our central lemma.

Lemma 4.1. If G is a connected graph, then $|V(G)|$-dimensional matching $\leq_p$ FACT(G).

Proof. Let p be any positive integer, and $U \subseteq \{1,2,\ldots,p\}^k$, where $k = |V(G)|$. It suffices to show how to construct (in polynomial time) a graph $R(U)$ with the property that $R(U)$ admits a G-factor if and only if U admits a k-dimensional matching.

$R(U)$ contains, among others, an independent set of pk vertices labelled by the pairs (i,j), where $1 \leq i \leq p$ and $1 \leq j \leq k$. For each k-tuple $(t_1,\ldots,t_k) \in U$, $R(U)$ contains a distinct copy of $S[G;v^*]$ whose k connector vertices are arbitrarily identified with the k vertices labelled $(1,t_1),\ldots,(k,t_k)$.

Suppose that U admits a k-dimensional matching W. We construct a G-factor φ of $R(U)$ as follows. To those copies of $S[G;v^*]$ associated with k-tuples in W, φ binds all of their vertices (in particular, their connector vertices). To all other copies of $S[G;v^*]$, φ binds only their interior vertices. Thus, φ binds the vertex (i,j) to the star module associated with that unique k-tuple in W containing j in position i. It follows that φ is a G-factor of $R(U)$.

Conversely, suppose that $R(U)$ admits a G-factor φ. We construct a k-dimensional matching W of U as follows. Call a copy of $S[G;v^*]$ in $R(u)$ "chosen" if φ binds to that copy all of its connector vertices. By Lemma 3.5 φ chooses exactly p of the star-modules in R(u). Let W be the set of k-tuples associated with chosen star modules. Since each vertex (i,j) is bound to exactly one chosen star module, it follows that exactly one element of W contains j in its i-th component. Hence W is a k-dimensional matching of U . □

We now restate and give a direct proof of our central result.

Theorem 4.2. If G is not of the form $\alpha \cdot K_1 \cup \beta \cdot K_2$ then FACT(G) is NP-complete.

Proof. It is clear that all problems FACT(G) are in NP. By lemma 2.1, it suffices to show that if G is a connected graph with at least three vertices, then FACT(G) is NP-complete. But this is immediate from Lemma 4.1 and the NP-completeness of k-dimensional matching, for $k \geq 3$. □

Thus virtually all uniform factoring problems (with the exception of matching) are NP-complete. A similar characterization holds for what we call "strict" G-factors.

A G-packing (or G-factor) of H is strict if each G_i belonging to the packing is an induced subgraph of H . Corresponding to FACT(G) we have the question S-FACT(G) expressed as:

INSTANCE: A graph H .

QUESTION: Does H admit a strict G-factor?

Note that H admits a strict G-factor if and only if its complement $\overline{H}$ admits a strict $\overline{G}$-factor. Then S-FACT(G) and S-FACT($\overline{G}$) are polynomially equivalent, and it is sufficient to consider problems

S-FACT(G) for connected graphs G only. Clearly, if G has fewer than three vertices a polynomial algorithm for S-FACT(G) follows from algorithms for maximum matching (eg. [7]). As with FACT(G) all other cases appear to be intractable.

<u>Theorem 4.3.</u> If G has at least three vertices then S-FACT(G) is NP-complete.

<u>Proof.</u> Observe that, in the proof of lemma 4.1., the graph $R(\bar{U})$ admits a G-factor if and only if it admits a strict G-factor. (This follows from the construction of star modules.) Hence, lemma 4.1 also proves that $|V(G)|$-dimensional matching $\leq_p$ S-FACT(G). Thus, the result follows from obvious fact that S-FACT(G) $\in$ NP and the NP-completeness of k-dimensional matching, for $k \geq 3$. □

5. Family Packings and Factors.

We introduced G-packings and G-factors as a generalization of conventional matchings and have reached the unfortunate conclusion that this is an unlikely direction in which to generalize the rich theory - most notably the existence of polynomial time bounded algorithms - that is associated with the matching problem. However, a straightforward extension of the notion of G-packing suggests itself as another natural generalization of matching. While negative results still abound, this extension does give rise to a number of positive results which hint at a new generalized theory of matching including both polynomial algorithms and elegant duality results.

We extend the notion of G-packing by replacing G by a family $\mathcal{G}$ of "packing" graphs. A $\mathcal{G}$-packing of a graph H is a set $\{G_1,\ldots,G_d\}$ of disjoint subgraphs of H such that each G_i is isomorphic to some element of $\mathcal{G}$. A $\mathcal{G}$-factor is defined similarly. The existence problem for $\mathcal{G}$-factors, denoted FACT($\mathcal{G}$), becomes:

INSTANCE: A graph H .

QUESTION: Does H admit a $\mathcal{G}$-factor:

As an example, if C_t denotes the cycle on t vertices and $\mathcal{G} = \{K_2, C_3, C_4, C_5, \ldots\}$, then FACT($\mathcal{G}$) can be solved as an assignment problem, [33].

The NP-completeness of many problems FACT($\mathcal{G}$) stems directly drom our earlier constructions. As a simple example, consider:

Example 5.1. FACT($\{K_t \mid t \geq 3\}$) is NP-complete.

Proof. It suffices to observe that the graph used in our reduction of 3-dimensional matching to FACT(K_3) contains no complete subgraphs of order four. Thus any $\{K_t \mid t \geq 3\}$-factor must also be a K_3-factor. □

It is interesting to note that H has a $\{K_t \mid t \geq 3\}$-factor if and only if its complement has a colouring in which each colour class contains at least three vertices. This connection with colouring is explored in more detail in [19]. Example 5.1 is subsumed by the following theorem whose proof will appear elsewhere.

Theorem 5.2. Let G be any subset of $\{K_t \mid t \geq 1\}$. If $K_1 \in G$ or $K_2 \in G$ then FACT(G) is in P, otherwise FACT(G) is NP-complete.

One further example should help to substantiate our claim that the study of family factorizations is a fertile setting in which to generalize the traditional theory of matching.

Example 5.3. FACT($\{K_{1,t} \mid t \geq 1\}$) is in P.

Proof. It is straightforward to confirm that a graph H admits a $\{K_{1,t} \mid t \geq 1\}$-factor if and only if it contains no isolated vertices.

The facility location (or domination number) problem [4, 11] can be viewed as trying to find a minimal $\{K_{1,t} \mid t \geq 1\}$-factor. Our framework makes it natural to express the related problem of determining the existence of factors using only a restricted subset of facilities (star graphs). Example 5.3 is just one special case of the following:

<u>Theorem 5.4.</u> Let G be any subset of $\{K_{1,t} \mid t \geq 1\}$. If for some $t \geq 1$, $K_{1,t} \notin G$ and $K_{1,t+1} \in G$, then FACT(G) is NP-complete. Otherwise FACT(G) is in P .

The proof of Theorem 5.4 also includes a duality result analogous to the theorems of Tutte [38] and Berge [1, p. 159] for star matchings; it will appear elsewhere. We have similar results for any set of complete bipartite graphs.

6. Conclusions.

We have shown that all uniform factorization problems, with the sole exception of matching, are NP-complete. While this result is of interest in its own right as a contribution to our knowledge of NP-completeness, it also lays the foundation as outlined in Section 5, for a new generalized theory of matching including new polynomial algorithms. The details of this theory will be explored elsewhere.

References

[1] C. Berge, Graphs and Hypergraphs, North-Holland, Amsterdam 1973.

[2] F.T. Boesch and J.F. Gimpel, "Covering the Points of a Digraph with Point-Disjoint Paths and its Application to Code Optimization". J. ACM 24 (1977), 192-198.

[3] N. Christofides, "Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem", GSIA, Carnegie-Mellon University, 1976.

[4] E. Cockayne, S. Goodman, and S. Hedetniemi, "A Linear Algorithm for the Domination Number of a Tree", Information Processing Letters 4 (1975), 41-44.

[5] S. A. Cook, "The Complexity of Theorem-Proving Procedures", Proc. Third ACM Symp. on Theory of Competing (1971), 151-158.

[6] W. H. Cunningham and A. B. Marsh, III, "A Primal Algorithm for Optimum Matching", Math. Programming Study 8 (1978), 50-72.

[7] J. Edmonds, "Paths, Trees, and Flowers", Canad. J. Math. 17 (1965), 449-467.

[8] J. Edmonds, "Maximum Matching and a Polyhedron with (0,1) Vertices", J. Res. Nat. Bureau of Standards 69B (1965), 125-130.

[9] J. Edmonds and E. L. Johnson, "Matching: a well-solved class of integer linear programs", in R.K. Guy et al., eds., Combinatorial Structures and their Applications, Gordon and Breach, N.Y., 1970, pp. 89-92.

[10] J. Edmonds and E. L. Johnson, "Matching, Euler Tours, and the Chinese Postman", Math. Programming 5(1973), 88-124.

[11] M. Farber, "Domination and Duality in Weighted Trees", Proc. 12th Southeastern Conf. on Combinatorics, Graph Theory and Computing, to appear.

[12] L. R. Ford, Jr. and D.R. Fulkerson, Flows on Networks, Princeton University Press, 1962.

[13] M. Fujii, T. Kasami and K. Ninamiya, "Optimal Sequencing of Two Equivalent Processors", SIAM J. Appl. Math. 17 (1969), 784-789, Erratum, ibid. 20 (1971), 141.

[14] M.R. Garey and D.S. Johnson, Computers and Intractability, W. H. Freeman and Company, San Fransisco, 1979.

[15] L. L. Garnishteyn, "The Partitioning of Graphs", Engineering Cybernatics, 1 (1969), 76-82.

[16] P.C. Gilmore and R.E. Gomory, "Sequencing a one-state variable Machine: A solvable Case of the Travelling Salesman Problem", Operations Research 12 (1964), 655-679.

[17] M. Hall, "Distinct representations of subsets", Bull. Amer. Math. Soc. 54 (1948), 922-926.

[18] F. Harary, Graph Theory, Addison-Wesley 1968.

[19] P. Hell and D.G. Kirkpatrick, "Scheduling, Matching and Coloring", in G.R. Szasz et al., eds., Algebraic Methods in Graph Theory, Colloq. Math. Soc. Janos Bolyai, 1981.

[20] P. Hell and D.G. Kirkpatrick, "On Generalized Matching Problems", Information Processing Letters, 12 (1981), 33-35.

[21] L.J. Herbert, "Some Applications of Graph Theory to Clustering", Psychometrika 39 (1974), 283-309.

[22] A.J. Hoffman and H.M. Markowitz, "A Note on Shortest Path, Assignment, and Transportation Problems", Naval Res. Logist. Quart. 10 (1963), 375-380.

[23] J.E. Hopcroft and R.M. Karp, "An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs", SIAM J. Comput. 2 (1973), 225-231.

[24] A.K. Hope, "Component Placing through Graph Partitioning in Computer-Aided Printed-Wiring-Board Design", Electronic Letters 8 (1972), 87-88.

[25] D.S. Johnson, private communication, August 1977.

[26] O. Kariv, An $O(n^{2.5})$ Algorithm for Finding Maximum Matching in a General Graph, Ph.D. Thesis, Weizmann Institute, Israel, 1976.

[27] R.M. Karp, Reducibility among Combinatorial Problems, in R.E. Miller and J.W. Thatcher, eds., Complexity of Computer Computations, Plenum Press, N.Y. 1972, pp. 85-103.

[28] R.M. Karp, "On the complexity of Combinatorial Problems", Networks 5(1975), 45-68.

[29] B.W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs", The Bell System Tech. J. 49 (1970, 291-307.

[30] D.G. Kirkpatrick and P. Hell, "On the Completeness of a Generalized Matching Problem", in Proc. Tenth Annual ACM Symposium on Theory of Computing, 1978, pp. 240-245.

[31] J.M. Klein and H. Takamori, "Parallel Line Assignment Problems", Mgt. Sci. 19 (1972), 1- 10.

[32] D. König, "Graphs and Matrices", Mat. Fiz. Lapok 38 (1931), 116-119.

[33] H.W. Kuhn, "The Hungarian Method for the Assignment Problem", Naval Res. Logist. Quart. 2 (1955), 83-97.

[34] E.L. Lawler, Combinatorial Optimization, Holt, Rinehart and Winston, N.Y., 1976.

[35] J.M. Lewis, "On the Complexity of the Maximum Subgraph Problem", Proc. Tenth Annual ACM Symposium on Theory of Computing, pp. 265-274.

[36] Mirsky, Perfect, Transversal Theory.

[37] J. Mühlbacher, "F-factors of graphs: a generalized matching problem, Information Processing Lett. 8 (1979), 207-214.

[38] T.J. Shaefer, "The Complexity of Satisfiability Problems, Proc. Tenth Annual ACM Symposium on Theory of Computing, pp. 216-226.

[39] W.T. Tutte, "The Factorisation of Linear Graphs", J. London Math. Soc. 22 (1947), 107-111.

[40] M. Yanakakis, "Node-and Edge-Deletion NP-Complete Problems" Proc. Tenth Annual ACM Symposium on Theory of Computing, pp. 253-264.