MIT/LCS/TM-273

GENERALIZED PLANAR MATCHING

Fran Berman

Tom Leighton

Peter Shor

Larry Snyder

April 1985

Generalized Planar Matching

Fran Berman[1]
Tom Leighton[2]
Peter W. Shor[2]
Larry Snyder[3]

[1] Computer Science Department
Purdue University
West Lafayette, Indiana 47907

[2] Mathematics Department and
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

[3] Computer Science Department
University of Washington
Seattle, Washington 98195

Abstract: In this paper, we prove that *maximum planar H-matching* (the problem of determining the maximum number of node-disjoint copies of the fixed graph H contained in a variable planar graph G) is NP-complete for any connected planar graph H with three or more nodes. We also show that *perfect planar H-matching* is NP-complete for any connected outerplanar graph H with three or more nodes, and is, somewhat surprisingly, solvable in linear time for triangulated H with four or more nodes. The results generalize and unify several special-case results proved in the literature. The techniques can also be applied to solve a variety of problems, including the optimal tile salvage problem from wafer-scale integration. Although we prove that the optimal tile salvage problem and others like it are NP-complete, we also describe provably good approximation algorithms that are suitable for practical applications.

Key Words: Approximation Algorithm, Covering, Matching, NP-Complete, Optimal Tile Salvage, Packing, Planar Graph, Wafer Scale Integration.

Fran Berman was supported by a Purdue Research Foundation Summer XL Grant and NSF Grant MCS-80-05387. Larry Snyder was supported by ONR Contract N00014-8-K-0360. Tom Leighton and Peter Shor were supported by Air Force contract OSR-82-0326, DARPA contract N00014-80-C-0622, and the Bantrell Foundation.

1. Introduction

Generalized matching problems have been studied in a wide variety of contexts. (See [6] and [8] for a large collection of references.) One common form of generalized matching is to find the maximum number of node-disjoint copies of some fixed graph H in a variable graph G. This form of the problem is called *maximum H-matching*. When H is an edge, this is simply the standard maximum matching problem. It is well-known that maximum matching can be solved in polynomial time. In [8], however, Kirkpatrick and Hell show that any non-trivial generalization of maximum matching is NP-complete. In fact, they show that *perfect H-matching* (the problem of deciding whether or not the nodes of a variable graph G can be completely covered by node-disjoint copies of a fixed graph H) is NP-complete for any connected H with three or more nodes. As a consequence, the maximum H-matching problem is also shown to be NP-complete for any connected graph H with three or more nodes.

In this paper we consider generalizations of the matching problem for *planar* graphs. In particular, we focus on the *maximum planar H-matching* and *perfect planar H-matching* problems, which are defined below. (Henceforth H and G are assumed to be connected planar graphs.)

Problem: Maximum Planar H-matching.
Instance: A planar graph G and an integer k.
Question: Does G contain k node-disjoint copies of H?

Problem: Perfect Planar H-matching.
Instance: A planar graph G.
Question: Can the nodes of G be completely covered with node-disjoint copies of H?

Although the Kirkpatrick-Hell result does not extend to planar graphs, a number of NP-completeness results have been proved for particular values of H. For example, maximum planar H-matching was recently shown to be NP-complete for $H = K_3$ (a triangle) and $H = K_{1,3}$ (a claw) [4].

No non-trivial graphs H were known for which either the maximum planar H-matching or the perfect planar H-matching problems could be solved in polynomial time. This fact, combined with the Kirkpatrick-Hell result (as well as other general NP-completeness results along these lines) leads one to conjecture that both problems are NP-complete for any connected H that contains three or more nodes. In Theorem 1 of Section 2, we prove that the conjecture is true for *maximum* planar H-matching. As we show in Section 3, however, the conjecture is false for *perfect* planar H-matching. In particular, we show that perfect planar H-matching is NP-complete for any connected *outerplanar* H with three or more nodes, but also that the problem is solvable in linear time for any *triangulated* H with four or more nodes. The precise characterization of H for which perfect planar H-matching is solvable remains a difficult and interesting open question.

In addition to proving NP-completeness results, we also consider approximation algorithms for maximum planar H-matching. In particular, we give a simple argument to show that there is a polynomial time algorithm which is guaranteed to find $(1-\epsilon)k$ node-disjoint copies of H in any planar bounded-degree graph G where k is the *maximum* number of node-disjoint copies of H in G and $\epsilon = O(1/\sqrt{\log k})$. (In [1], Baker describes a more general and probably more practical algorithm that works for all planar graphs. No such results are known for the non-planar version of this problem.) Moreover, we show that it is unlikely that a substantially better approximation algorithm exists. In fact, we show that if P$\neq$NP, then there is no polynomial-time $(1-\epsilon)$-approximation algorithm where $\epsilon = O(1/k^{\alpha})$ for any $\alpha > 0$.

The techniques developed in this paper can be applied to solve a variety of related matching problems. For example, the problem which originally motivated us to study generalized planar matching comes from wafer-scale integration. This problem is known as the *optimal tile salvage problem* [2] and consists of finding the maximum number of non-overlapping 2×2 regions of functioning cells in a $\sqrt{N} \times \sqrt{N}$ grid of cells, some of which are faulty. Although Fowler, Paterson and Tanimoto [5] proved that finding the maximum number of 3×3 squares of functioning cells is NP-complete, the complexity of the 2×2 problem remained unknown. In Section 4, we apply the techniques developed in the paper to give a simple proof that the $x \times y$ optimal tile salvage problem is NP-complete for all $\{x, y\}$ except $\{1, 1\}$ and $\{1, 2\}$. Fortunately, we are also able to provide a simple, fast and efficient approximation algorithm for this problem.

The techniques developed in this paper can also be applied to problems for which the copies of H in G must only be *edge-disjoint* or for which the copies of H must be *induced* in G, although we have not worked out the details in this paper. They also appear to be useful for reductions to minimum covering problems, planar packing problems, three-dimensional packing problems, and certain planar games like "dots and boxes." [2-8]

The remainder of the paper is divided into five sections. Section 2 contains our results on maximum planar H-matching. The perfect planar H-matching results are in Section 3. Applications of our techniques are described in Section 4. We conclude with acknowledgements and references in Sections 5 and 6.

2. Maximum Planar H-Matching

In this section, we determine the complexity of maximum planar H-matching for any H. Without loss of generality, we assume that H and G are connected. (It is easy to extend the results to H and G that are not connected.) If H has two or fewer nodes, then the problem is easy. In Section 2.1, we show that maximum planar H-matching is NP-complete for any H with three or more nodes. We describe approximation algorithms for this problem in Section 2.2.

2.1 NP-Completeness

Theorem 1: *Maximum Planar H-matching is NP-complete for any H with three or more nodes.*

Proof: The analysis is divided into two classes depending on whether or not the largest 2-connected component in H is unique. (Cut edges are considered to be degenerate examples of 2-connected components.) A special case is considered for each class before the result is proved. For H that contain a unique maximum-size 2-connected component, the special case is a cycle with 3 nodes. For H that contain two or more maximum-size 2-connected components, the special case is a path with 3 nodes. In all cases, the reduction is from Planar 3-SAT [9].

Class 1: *H contains 3 or more nodes and a unique maximum-size 2-connected component.*

Proof for Special Case: *H is a 3-cycle.*

Given a planar 3-SAT problem P with variables $x_1, ..., x_r$ and clauses $c_1, ..., c_s$, let $G(P)$ be the associated planar graph with nodes $x_1, ..., x_r, c_1, ..., c_s$ and edges representing an incidence of a variable in a clause. In what follows, we will show how to transform $G(P)$ into another planar

Given a planar 3-SAT problem P with variables $x_1, ..., x_r$ and clauses $c_1, ..., c_s$, let $G(P)$ be the associated planar graph with nodes $x_1, ..., x_r, c_1, ..., c_s$ and edges representing an incidence of a variable in a clause. In what follows, we will show how to transform $G(P)$ into another planar graph $G^*(P)$ that has $rs + s$ disjoint 3-cycles precisely when P is satisfiable. As a result, we will have shown that maximum planar H-matching is NP-complete in the special case that H is a 3-cycle.

To construct $G^*(P)$, we will replace every variable node of $G(P)$ with a *generator* (see Figure 1) and every clause node with a *receptor* (see Figure 2). Each generator consists of $4s$ nodes, $2s$ of which are designated as *connection nodes.* Connection nodes appear as empty circles and (in Figure 1) are divided into consecutive pairs. The first node in each pair is called a *positive* connection node and the second node is called a *negative* connection node. Each receptor has 5 or 7 nodes (2 or 3 of which are connection nodes) depending on whether the corresponding clause contains 2 or 3 variables. Without loss of generality, we assume that every variable appears in each clause at most once.

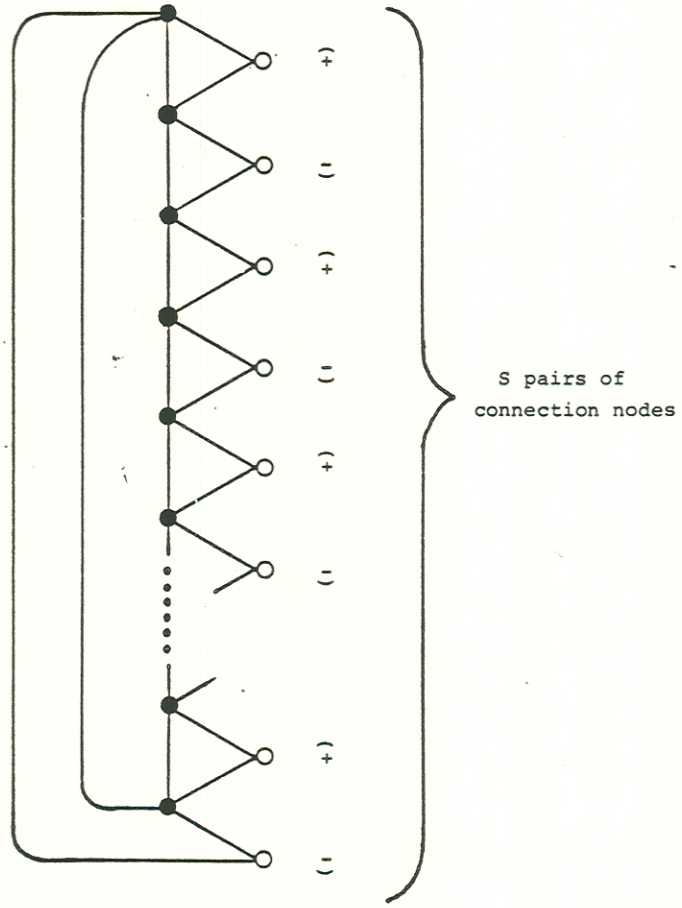

Figure 1: *Generator for a 3-cycle.*

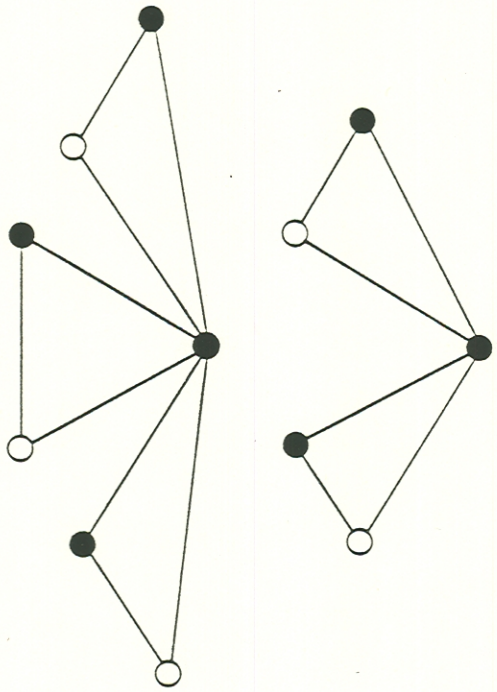

Figure 2: *Receptors for a 9-cycle.*

The embedding of $G(P)$ in the plane provides a cyclic ordering of the edges around each vertex. Pick a linear ordering of the edges around each vertex which is consistent with the cyclic ordering. If variable x_i appears uncomplemented in clause c_j, and the edge from x_i to c_j in $G(P)$ is in the pth position of the linear ordering at x_i and in the qth position of the ordering at c_j, then identify the pth *positive* connection node of the ith generator with the qth connection node of the jth receptor. If variable x_i appears complemented in clause c_j, then identify the pth *negative* connection node of the ith generator with the qth connection node of the jth receptor. Because $G(P)$ is planar, $G^*(P)$ must also be planar. For example, we have constructed $G(P)$ and $G^*(P)$ for the function $(x_1 + x_2 + x_3)(\bar{x}_1 + \bar{x}_3 + x_4)(\bar{x}_2 + x_3 + \bar{x}_4)$ in Figures 3a and 3b.

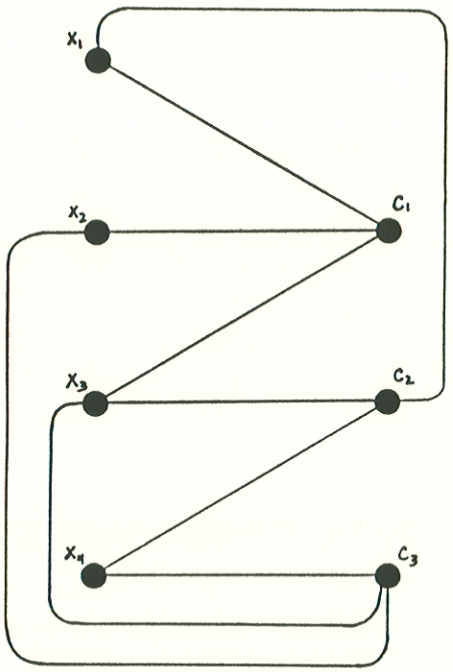

Figure 3a:

$G(P)$ *for* $P = (x_1 + x_2 + x_3)(\bar{x}_1 + \bar{x}_3 + x_4)(\bar{x}_2 + x_3 + \bar{x}_4)$.

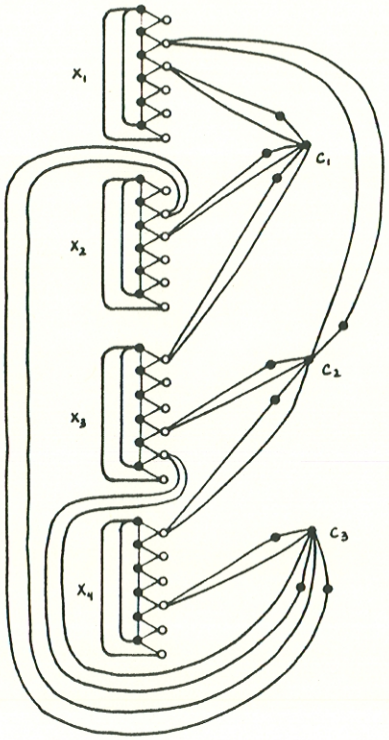

Figure 3b:

$G^*(P)$ *for* $P = (x_1 + x_2 + x_3)(\bar{x}_1 + \bar{x}_3 + x_4)(\bar{x}_2 + x_3 + \bar{x}_4)$.

It is easily seen that every 3-cycle contained in $G^*(P)$ is contained wholly within a generator or a receptor. In addition, each generator can contain at most s node-disjoint 3-cycles and each receptor can contain at most one 3-cycle. Hence the number of disjoint 3-cycles in $G^*(P)$ is at most $rs + s$. In fact, we will show that $G^*(P)$ contains $rs + s$ disjoint 3-cycles precisely when P is satisfiable, thus concluding the reduction.

As is shown in Figure 4, there are precisely two ways that a generator can contain s disjoint 3-cycles. One way (the true mode) requires the use of all the negative connection nodes but does not use any of the positive connection nodes. The other way (the false mode) requires the use of all the positive connection nodes but none of the negative connection nodes.

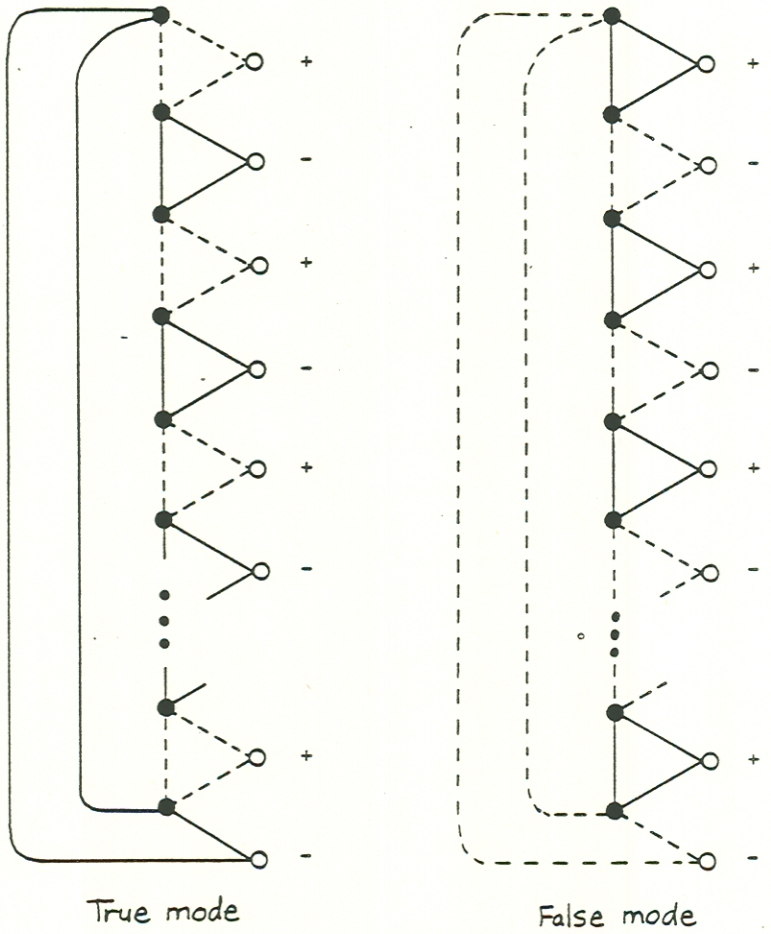

Figure 4: *Generators in true and false modes. (Solid lines denote edges used to form 3-cycles.)*

A receptor can contain a 3-cycle if and only if one of its connection nodes has been identified with a positive connection node of a generator in true mode or with a negative connection node of a generator in false mode. Thus, P is satisfiable if and only if there is a matching in which every receptor contains a 3-cycle. This concludes the proof that $G^*(P)$ contains $rs + s$ disjoint 3-cycles if and only if P is satisfiable.

Proof for General H in Class 1

Let H be any connected planar graph with at least 3 nodes for which the maximum-size 2-connected component in H (denoted by $\bar{H}$) is unique. Embed H in the plane so that the outer face is a cycle in $\bar{H}$. Identify any three of the nodes in the cycle as a, b, and c. For example, see Figure 5.

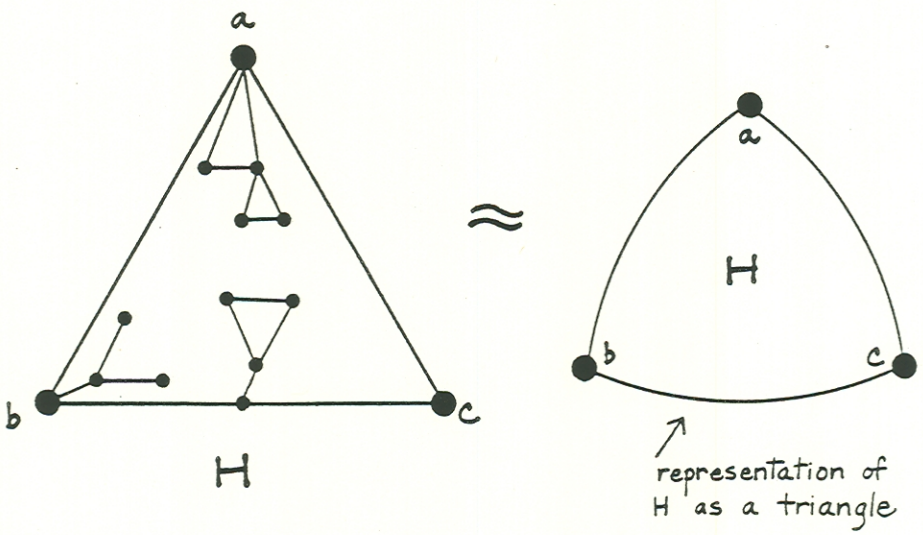

Figure 5: *Representation of an arbitrary class 1 graph as a 3-cycle.*

Notice that the embedding of H shown in Figure 5 looks very much like a 3-cycle with vertices a, b and c. Using this similarity, generators and receptors can be constructed as shown in Figures 6 and 7.

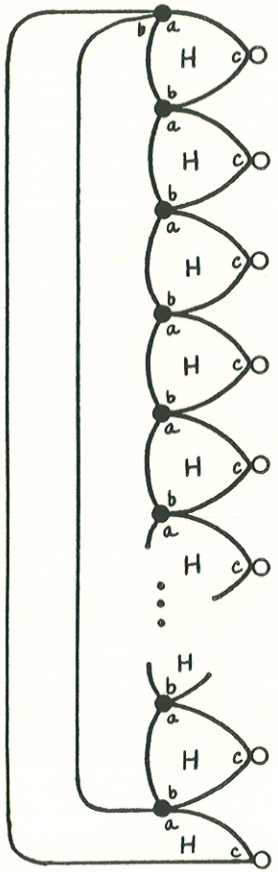

Figure 6: *Generator for class 1.*

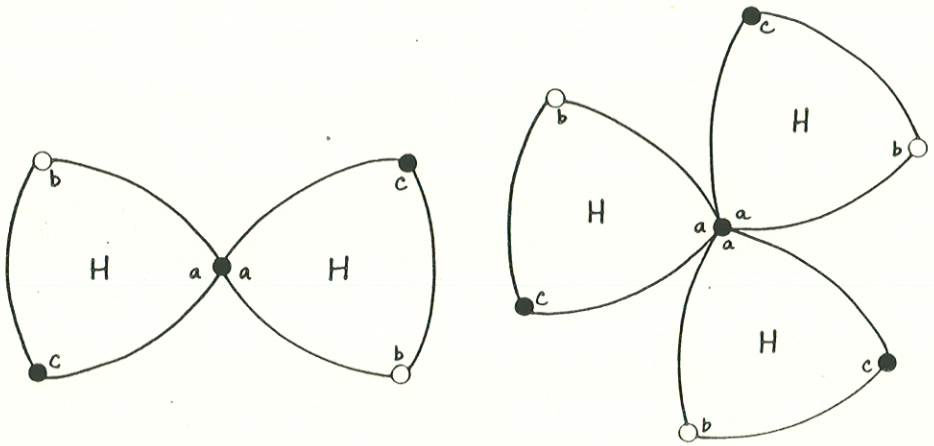

Figure 7: *Receptors for class 1.*

Given a planar 3-SAT problem P, the construction of $G^*(P)$ is the same as before except that the generators and receptors in Figures 6 and 7 are used in place of those in Figures 1 and 2. As before, $G^*(P)$ is planar and contains $rs + s$ disjoint copies of H if P is satisfiable. It remains only to show that $G^*(P)$ contains less than $rs + s$ disjoint copies of H if P is not satisfiable.

Since every copy of H contains a copy of $\bar{H}$, the number of disjoint copies of $\bar{H}$ in $G^*(P)$ is an upper bound on the number of disjoint copies of H in $G^*(P)$. In what follows we will show that $G^*(P)$ contains less than $rs + s$ copies of $\bar{H}$ if P is not satisfiable, thus concluding the proof.

There are two ways that a copy of $\bar{H}$ can occur in $G^*(P)$: totally within a triangular copy of H or across several triangular copies of H. Any copy of $\bar{H}$ which is contained within a triangular copy of H must utilize the a, b and c nodes in that copy of H since H contains only one maximum-size 2-connected component. Because $\bar{H}$ is 2-connected, any copy of $\bar{H}$ which spans several triangular copies of H must contain a cycle that contains several of the nodes labeled a, b or c in the various copies. Such a cycle may be formed with or without the use of receptor nodes but must always use two of the a, b and c nodes of each triangular copy of H that is entered by the cycle. Inspection of Figures 6 and 7 reveals that one of these copies of H has just two connections to the rest of $G^*(P)$. (In proof, note that receptor H's only have two connections to the rest of $G^*(P)$. Cycles not entering receptor triangles of H must enter every triangle of H in some generator. In every generator, there are always some triangular H not connected to a receptor.) Since the remainder of the nodes in this copy of H cannot be used to form other copies of $\bar{H}$ (they become disconnected from $G^*(P)$), the number of copies of $\bar{H}$ is not decreased by replacing the copy of $\bar{H}$ spanning several triangular copies of H with a copy of $\bar{H}$ contained entirely within the triangular copy of H having just two connections to the rest of $G^*(P)$. Hence every copy of $\bar{H}$ can be assumed to use the a, b and c nodes of some triangular copy of H. The remainder of the analysis is then identical to that for the special case of 3-cycles.

Class 2: *H contains two or more maximum-size 2-connected components.*

Proof for Special Case: *H is a path with three nodes.*

The proof is very similar to that for 3-cycles except that different generators and receptors are used. The new generators and receptors are displayed in Figures 8 and 9. As before, each generator has $2s$ connection nodes (recall that s is the number of clauses in the problem P) and each receptor has 2 or 3 connection nodes. The $2s$ connection nodes are divided into consecutive pairs of positive and negative nodes and the identification of nodes to form $G^*(P)$ is identical to that done for 3-cycles.

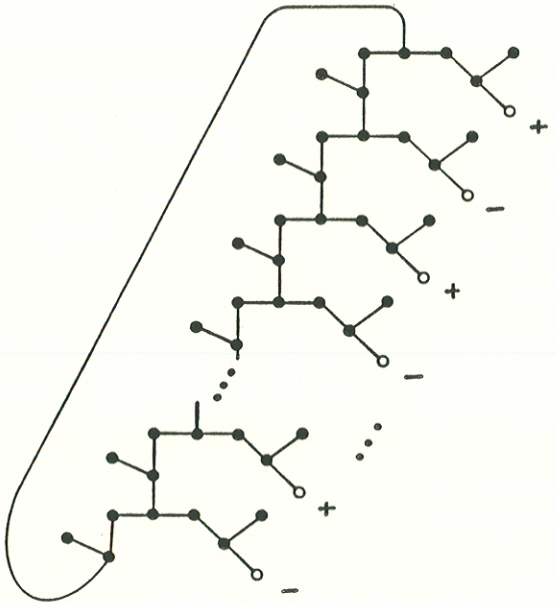

Figure 8: *Generator for 3-node path.*

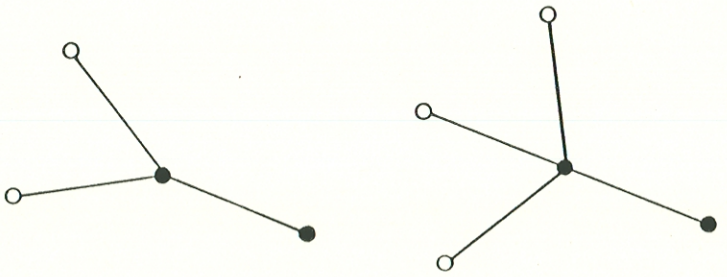

Figure 9: *Receptors for 3-node path.*

Each generator is formed by cascading $2s$ copies of the 9-node *gadget* shown in Figure 10. Without loss of generality, we can assume that all disjoint 3-node paths are contained totally within a receptor or a gadget like that shown in Figure 10. This is due to the fact that any 3-node path between 2 gadgets (or a gadget and a receptor) isolates a leaf in the neighboring gadget (or receptor). Since the leaf cannot be used by any other 3-node path, it might as well be used by the offending 3-node path in a way which places the 3-node path totally within the gadget or receptor. For example, see Figure 11.

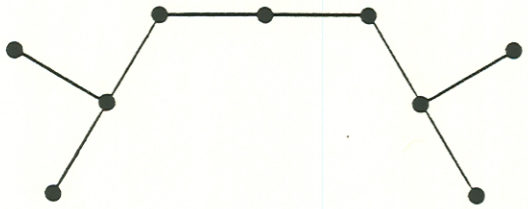

Figure 10: *Gadget used to build generators.*

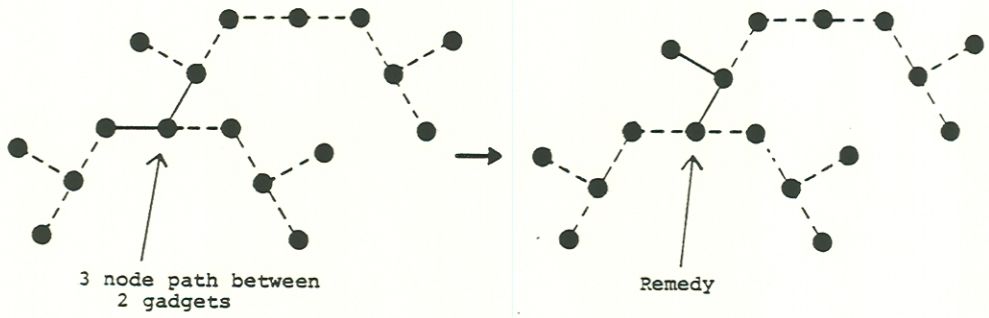

Figure 11: *Without loss of generality, every 3-node path is entirely contained within a gadget of a generator or within a receptor.*

It is easily seen that a gadget contains three disjoint 3-node paths. If one gadget is used to produce three disjoint 3-node paths in $G^*(P)$, however, its neighboring gadgets can only be used to produce two disjoint 3-node paths. (The neighboring gadgets would then have less than 9 nodes available.) Hence each generator contains at most $5s$ disjoint 3-node paths. There are only two ways to fit $5s$ disjoint paths in a generator. One way (the true mode) requires the use of all the negative connection nodes but does not use any of the positive connection nodes. The other way (false mode) requires the use of all of the positive connection nodes but none of the negative connection nodes. These two modes are illustrated in Figure 12.

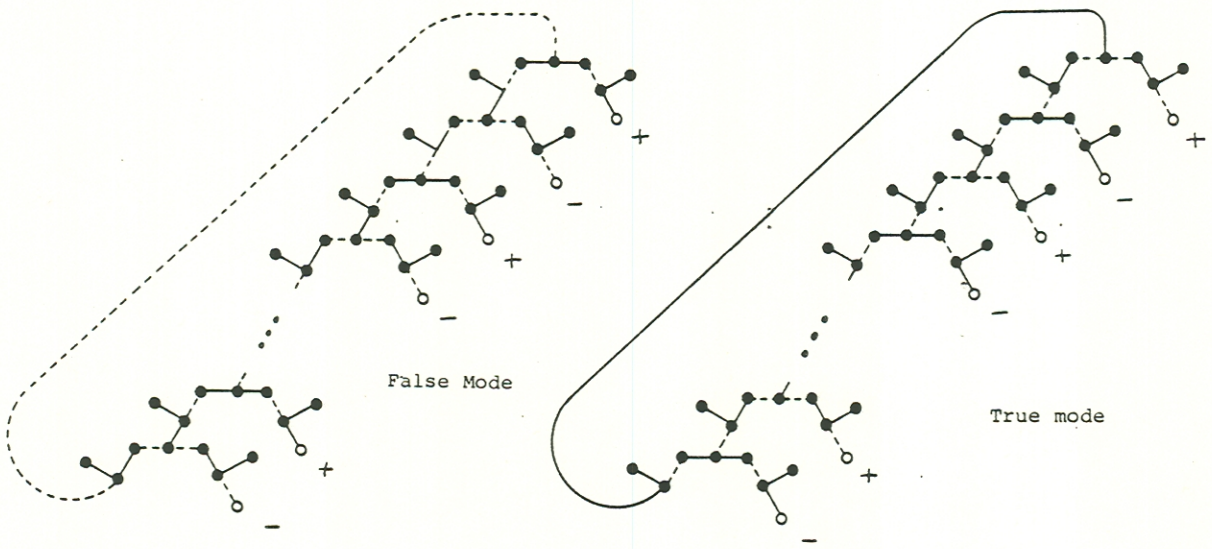

Figure 12: *True and false modes for generators. (Solid lines denote edges used to form 3-node paths.)*

A receptor can contain a 3-node path if and only if one of its connection nodes has been identified with a positive connection node of a generator which is in true mode or if one of its connection nodes has been identified with a negative connection node of a generator in false mode. Hence every receptor can contain a 3-node path if and only if P is satisfiable. Thus $G^*(P)$ contains $5rs + s$ disjoint 3-node paths if and only if P is satisfiable.

Proof for General H in Class 2

Let H be any connected planar graph containing two or more maximum-size 2-connected components. Find a cutpoint v of H which is contained in one of the maximum-size 2-connected components $\bar{H}$, and which separates it from the rest of the maximum-size 2-connected components in H. Let B denote the union of $\{v\}$ and the connected component of $H-\{v\}$ containing $\bar{H}-\{v\}$, and let A denote the union of $\{v\}$ and the rest of H. Further, let b be a node on the same face as v in $\bar{H}$, and let a be a node which is on the same face as v in A and which is in a 2-connected

component containing v that separates $\bar{H}$ from another maximum-size 2-connected component. If a maximum-size component in A contains v then choose a to be a node on the same face as v in that maximum-size component. For example, see Figure 13.

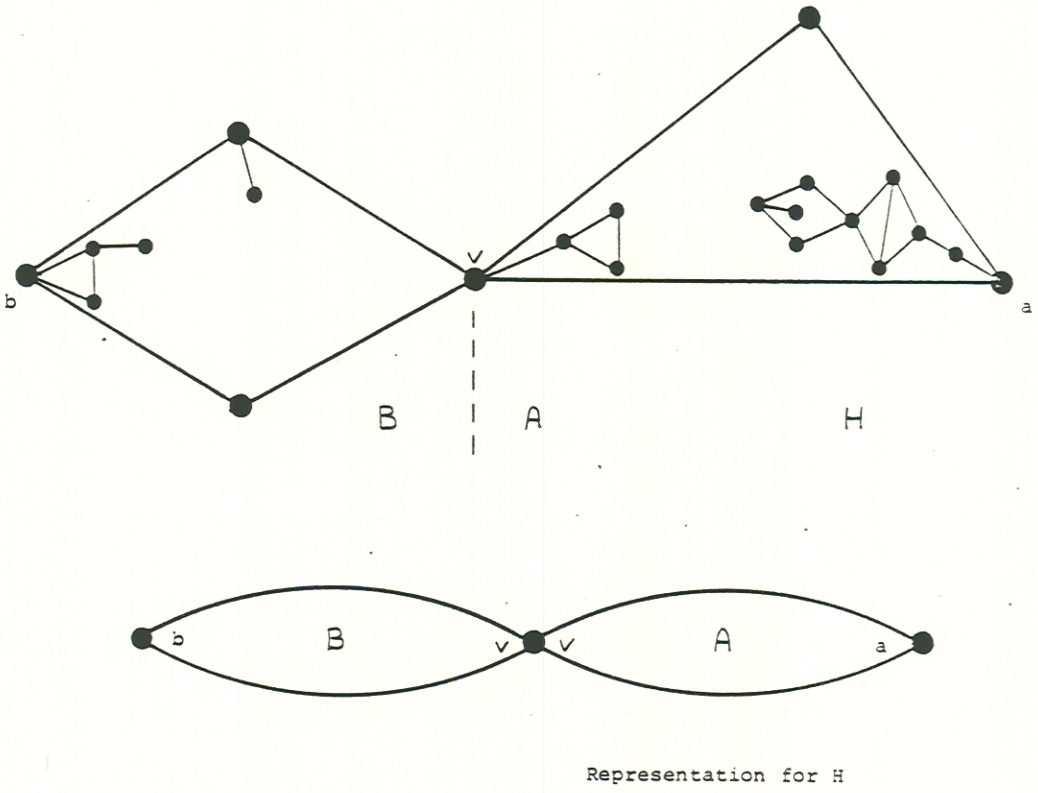

Figure 13: *Representation of an arbitrary class 2 graph as a 3-node path.*

As is shown in Figure 13, H has a planar embedding that resembles a path with 3 nodes (a, v, and b) and 2 edges (A and B). Using this analogy, it is possible to generalize the generators and receptors of Figures 8 and 9 as shown in Figures 14 and 15. Using the generalized generators and receptors, $G^*(P)$ is then constructed as before.

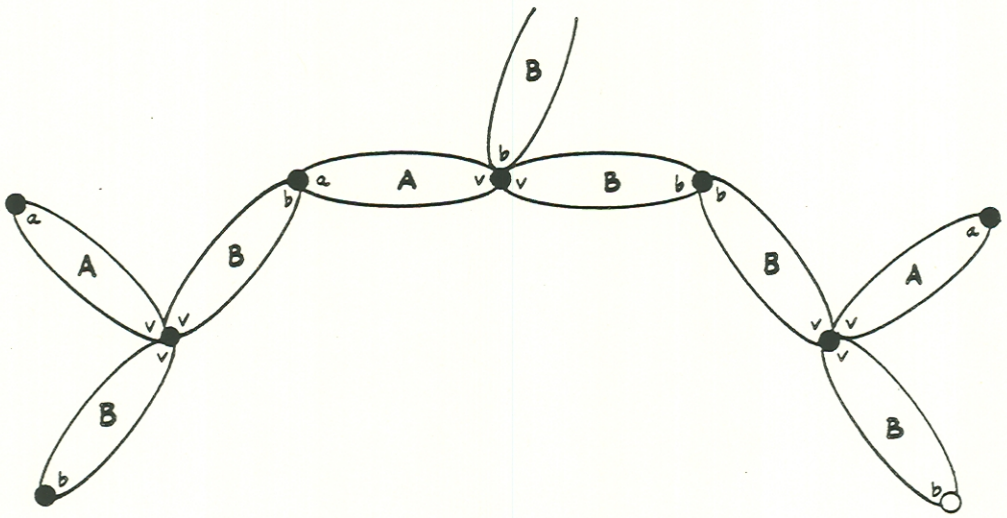

Figure 14: *Part of a generator for Class 2.*

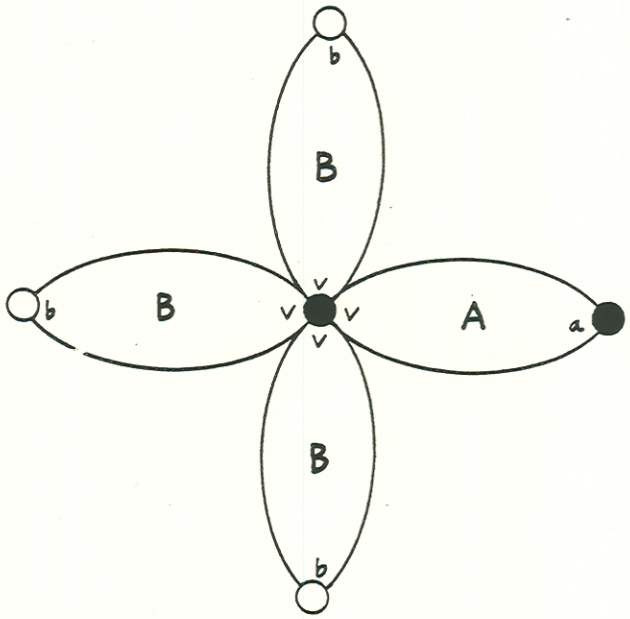

Figure 15: *One kind of receptor for Class 2.*

By previous arguments, it is clear that $G^*(P)$ contains $5rs + s$ disjoint copies of H if P is satisfiable. It remains to show that $G^*(P)$ cannot contain $5rs + s$ disjoint copies of H if P is not satisfiable. To prove this, we will show that (without loss of generality) every copy of H uses 3 of the a, b or v nodes in $G^*(P)$. By the arguments used for the 3-node path case, less than $5rs + s$ such objects can be contained in $G^*(P)$ if P is not satisfiable, thus concluding the proof.

Any copy of H which utilizes two or fewer of the a, b or v nodes must, without loss of generality, occur in one of the three ways shown in Figure 16. (Recall that if a copy of H utilizes a v node which corresponds to a "leaf" copy of A, then it might as well use the corresponding a node since that a node cannot be used by any other copy of H.)

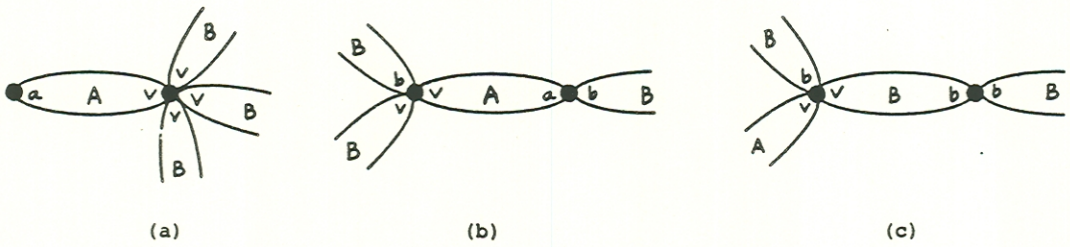

Figure 16: *Possible cases for occurrences of H in $G^*(P)$.*

In the first two cases (Figures 16a and 16b), there is one less copy of $\bar{H}$ available than is needed to make a copy of H. Hence a copy of H cannot occur in this fashion. In the last case (Figure 16c) there are just enough maximum-size 2-connected components available (provided that a is not in a maximum-size 2-connected component of H; otherwise we are done) but there are not enough nodes available which are in 2-connected components separating maximum-size 2-connected components. This is because all such nodes in H are contained in A and cannot be replaced by nodes in B. Since a is such a node, it must be used in order to complete a copy of H. Hence H cannot be contained in any of the structures shown in Figure 16 and the proof is complete. ∎

2.2 Approximation Algorithms

In [1], Baker describes a provably good approximation algorithm for a variety of problems including maximum planar H-matching. In what follows, we describe a simpler, but possibly less practical, approximation algorithm that works for planar graphs with bounded node degree. The algorithm is based on the Lipton–Tarjan planar separator theorem [10], runs in $O(N^{1+\delta})$ steps for any $\delta > 0$ and any N-node graph G, and is guaranteed to find at least $(1-\epsilon)k$ node-disjoint copies of H where k is the maximum number of node-disjoint copies of H in G and $\epsilon = O(1/\sqrt{\log k})$. Afterward, we show that any substantial improvement of this algorithm is unlikely. In fact, we prove in Theorem 3 that if P$\neq$NP, then there is no polynomial-time $(1 - \epsilon)$-approximation algorithm for maximum planar H-matching where $\epsilon = O(1/k^{\alpha})$ for any $\alpha > 0$.

The approximation algorithm consists of four steps, as follows.

Step 1: Remove nodes of G that are not contained in any copy of H. Call the resulting graph G' and let N' denote the number of nodes in G'.

Step 2: Repeatedly use the Lipton–Tarjan planar separator algorithm to partition G' into disconnected blocks, each with at most $\delta \log N'$ nodes where $\delta > 0$. At most $O(N'/\sqrt{\delta \log N'})$ edges are removed from G' in this process.

Step 3: By exhaustive search, find the maximum number of node-disjoint copies of H in each block of the partition.

Step 4: Output the union of the node-disjoint copies of H found in Step 3.

Theorem 2: *The preceding algorithm runs in $O(\frac{N^{1+\delta}}{\delta \log N})$ steps and finds $(1-\epsilon)k$ node-disjoint copies of H in G, where k is the maximum number of node-disjoint copies of H in G and $\epsilon = O(1/\sqrt{\delta \log k})$.*

Proof: Step 1 takes $O(N)$ steps since for each node there are at most a constant number of configurations of its neighbors that could form a copy of H. (Recall that G is assumed to have bounded node degree, and that H is fixed.) Step 2 takes $O(N' \log N') = O(N \log N)$ steps since we are applying the linear-time Lipton–Tarjan algorithm once for an N'-node graph, twice for $\frac{N'}{2}$-node graphs, and so forth for $O(\log N')$ iterations. Step 3 takes $\frac{N'}{\delta \log N'} O(2^{\delta \log N'}) = O(\frac{N^{1+\delta}}{\delta \log N})$ steps. Hence the running time is $O(\frac{N^{1+\delta}}{\delta \log N} + N \log N)$. For constant $\delta > 0$, this is $O(\frac{N^{1+\delta}}{\delta \log N})$.

Let k denote the maximum number of node-disjoint copies of H in G. At most $O(N'/\sqrt{\delta \log N'})$ copies of H contain an edge removed in Step 2. Hence at least $k - O(N'/\sqrt{\delta \log N'})$ copies of H are output in Step 4. Since every node of G' is contained in some copy of H and since every copy of H can overlap at most a constant number of other copies of H, we know that $k = \Omega(N')$. Hence $k - O(N'/\sqrt{\delta \log N'}) \geq k - O(k/\sqrt{\delta \log k})$ and the algorithm outputs at least $(1-\epsilon)k$ node-disjoint copies of A where $\epsilon = O(1/\sqrt{\delta \log k})$. ∎

Theorem 3: *If P≠NP, then there is no polynomial-time $(1-\epsilon)$-approximation algorithm for maximum planar H-matching when $\epsilon = 1/k^{\alpha}$ for any $\alpha > 0$, where k is the maximum number of node-disjoint copies of H in the input graph G, and H is any connected graph with three or more nodes.*

Proof: Fix $\alpha > 0$. We show that if a polynomial-time $(1 - 1/k^{\alpha})$-approximation algorithm existed, then it would be possible to construct an exact algorithm for maximum planar H-matching, which we showed was NP-complete in Theorem 1. Consider an N-node graph G and the question of whether or not G contains k node-disjoint copies of H. Construct a graph G' that consists of $T > k^{\frac{1-\alpha}{\alpha}}$ disconnected copies of G. If G contains at least k node-disjoint copies of H, then G' contains at least Tk node-disjoint copies of H. Otherwise, G' contains at most $T(k-1) = Tk - T$ node-disjoint copies of H. In the former case, a $(1 - 1/k^{\alpha})$-approximation algorithm finds $(1 - 1/(Tk)^{\alpha})Tk = Tk - (Tk)^{1-\alpha} > Tk - T$ node-disjoint copies of H. This cannot happen in the latter case. Hence, the approximation algorithm can be transformed into an exact algorithm. ∎

3. Perfect Planar H-Matching

In this section, we determine the complexity of perfect planar H-matching for large classes of H. In Section 3.1, we show the problem is NP-complete for connected outerplanar H with three or more nodes. In Section 3.2, we describe a linear time algorithm for perfect planar H-matching for triangulated H with four or more nodes. The precise characterization of H for which the problem is solvable in polynomial time remains an interesting open question.

3.1 NP-Completeness

Theorem 4: *Perfect planar H-matching is NP-complete for all outerplanar graphs H with at least 3 vertices.*

Proof: We use a reduction from 1-in-3 SAT. The plan of attack is as follows: Given a 1-in-3 SAT problem P, we take the clauses and the variables to be points in the plane, and make a graph $G(P)$ by connecting each clause to the variables it contains (we allow edges to cross). Next, we replace each edge by a *transmission line*, each variable with a *generator*, each clause with a *receptor* and each point where two edges cross with a *crossing mechanism*. We also attach transmission lines to unused connection nodes on generators. These "loose" transmission lines will then be brought together in *endings*, so as to be able to use all vertices of $G^*(P)$ given a satisfiable P. For example, Figure 17 shows a typical graph $G(P)$ before the nodes, edges, crossings and loose ends are replaced by the gadgets to form $G^*(P)$.

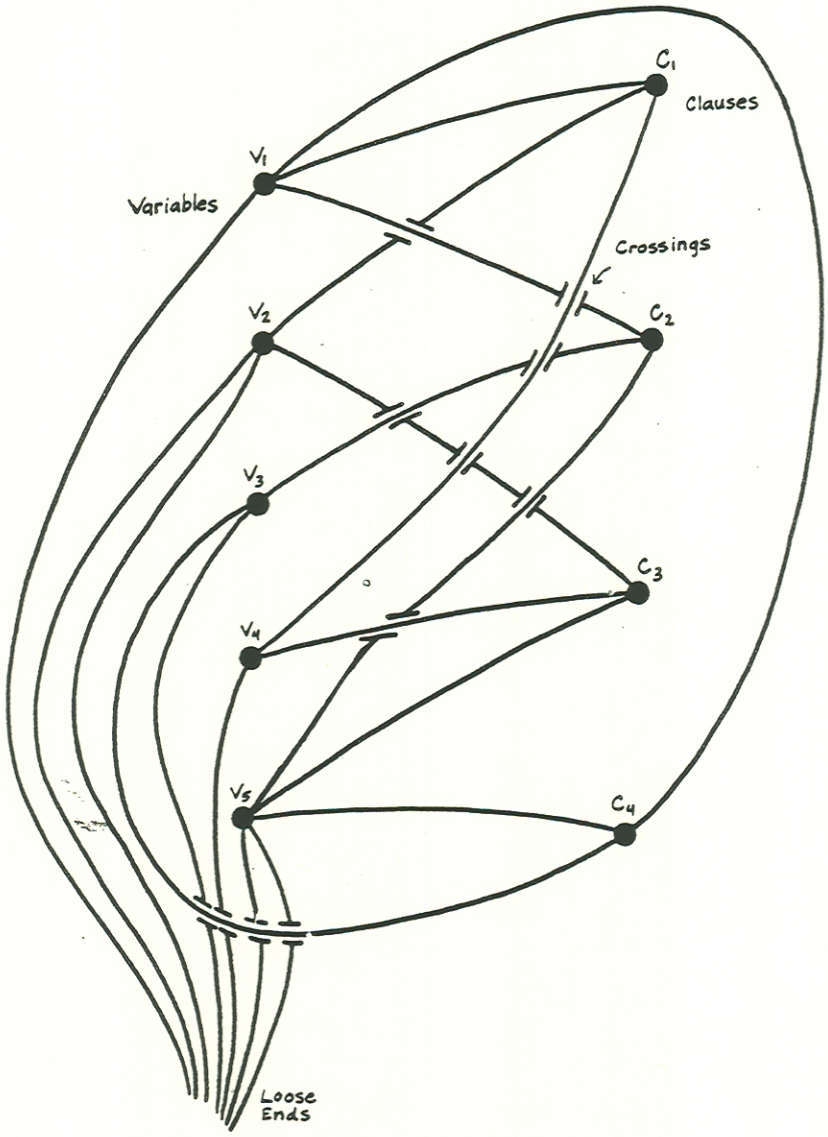

Figure 17: *A typical graph $G(P)$.*

For each gadget, there are two cases to consider depending on the degree of the minimum degree node v in H. Since H is outerplanar, it is easy to see that the degree of v is either 1 or 2. If the degree of v is 1, we say we are in *case 1*, and if it is 2, *case 2*. Let k be the number of vertices in H. Then, deleting v, we have a subgraph H^* with $k-1$ vertices which is connected to v by 1 or 2 edges (Figure 18). In most of our constructions, case 1 and case 2 are similar enough we need only illustrate one of them.

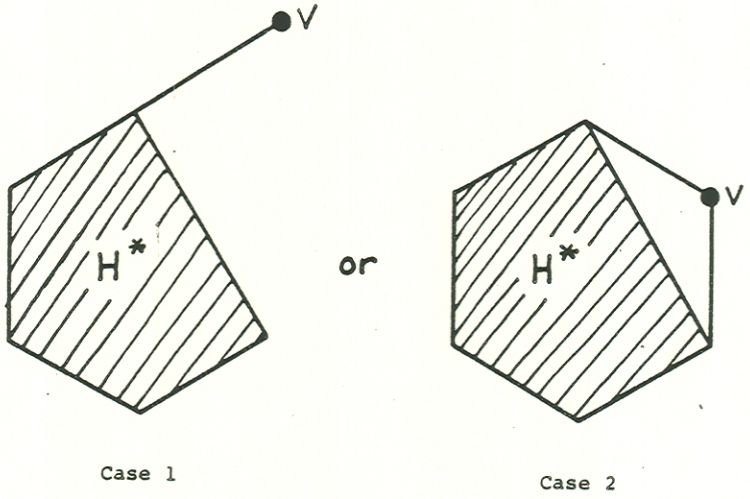

Figure 18: *The two cases for H.*

Transmission lines (Figure 19)

A transmission line can either be in true mode or false mode. In a perfect planar matching, since every vertex must be used, each vertex v must lie either to the left of the H^* to which it is attached (false) or to the right (true).

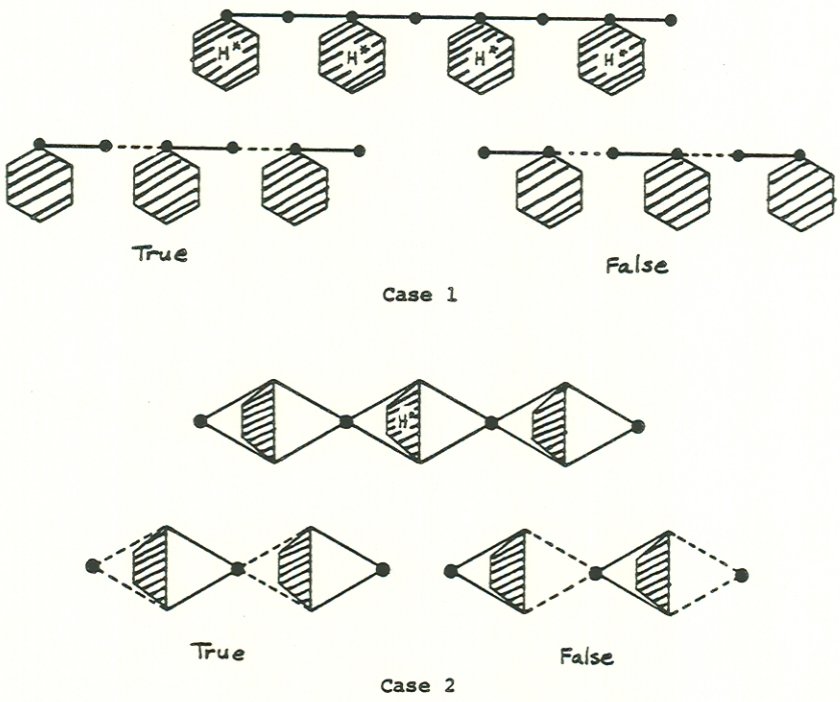

Figure 19: *Transmission lines.*

Inverters (Figure 20)

To construct generators, we need inverters. An inverter is a copy of H with transmission lines connected to every point (this is where we need H outer-planar). It has one signal going in, and $k-1$ signals going out, each of which is inverted. If the copy of H in the center is present in the matching, then the input is F and the outputs are T, and if it is absent, the input is T and the outputs F.

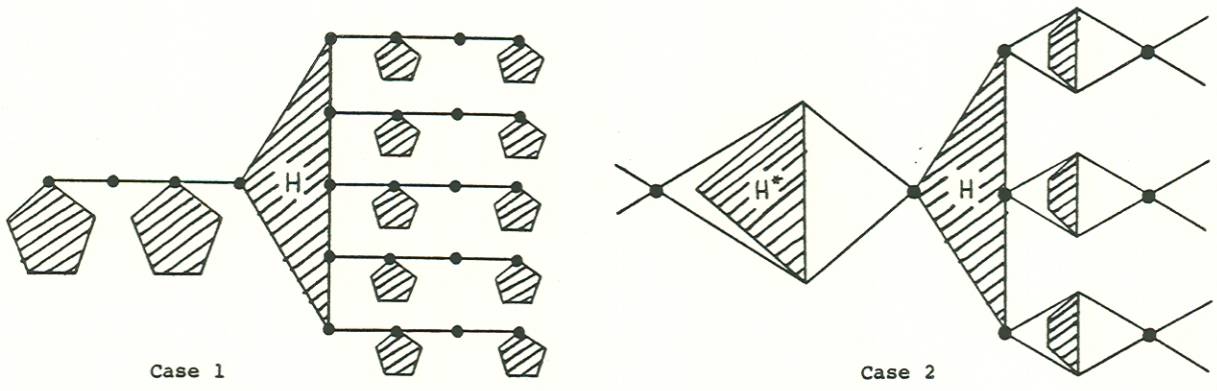

Figure 20: *Inverters*

Generators (Figure 21)

To make a generator, we first notice that a transmission line viewed backwards has the opposite value, so to obtain one output x and one $\bar{x}$ we need only take a transmission line and bend it. By adding inverters we can generate as many extra transmission lines for x and $\bar{x}$ as we want (since an inverter also increases the number of signals) although we could have to generate up to $2k$ more than we want. This is because each inverter gives us $k-1$ additional x's or $\bar{x}$'s.

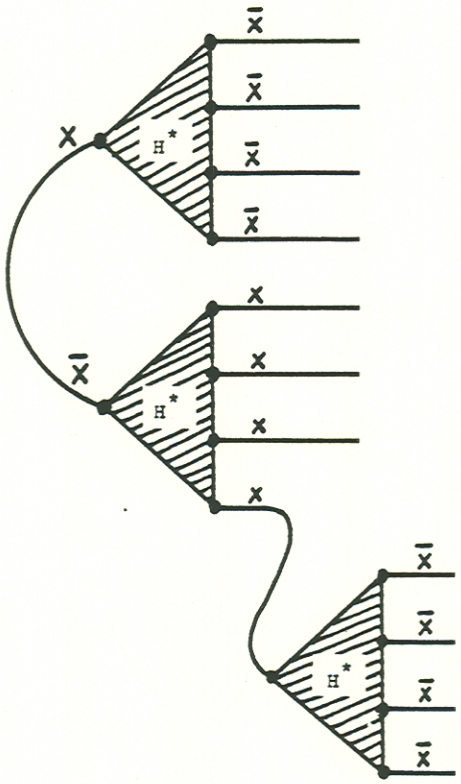

Figure 21: *Generator for case 1.*

Receptor (Figure 22)

A receptor is simply 3 transmission lines meeting at a point. In a perfect matching, one line must be true and the others false, since the meeting point must be covered by exactly one copy of H.

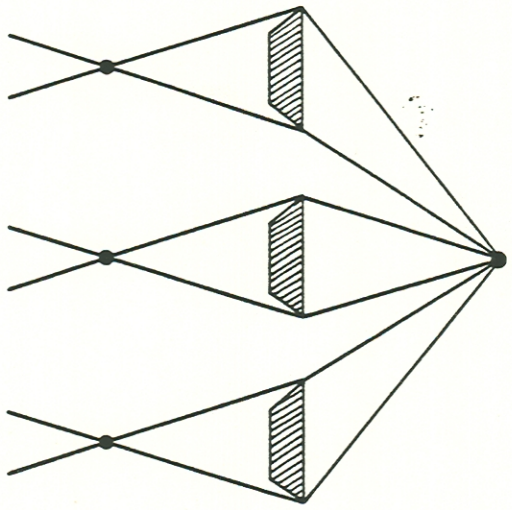

Figure 22: *Receptor for case 2.*

Branching (Figure 23)

To make crossing mechanisms and to tie up the loose transmission lines, we need a branching. This is like a receptor, except it has one input and several outputs. If the input is true, all of the outputs must be true. If it is false, one of the outputs will be false and the rest true. If you run a branching in reverse, with two inputs and one output, both inputs cannot be T in a perfect H-matching. If one input is T, and the other F, then the output is T. If both are F, the output is F. Similarly, if an inverter is reversed, the condition is imposed that all inputs are equal, and the output is inverted.

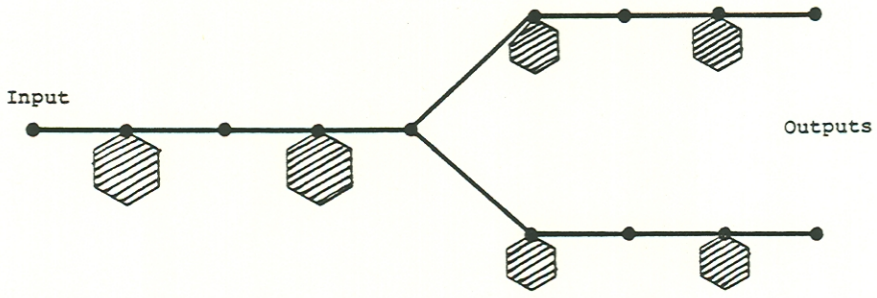

Figure 23: *A branching for case 1.*

Partial Crossing Mechanism (Figures 24 and 25)

We first devise a partial crossing mechanism so we can later build a crossing mechanism with it. This gadget accepts input signals which are (T, T), (T, F) or (F, T), and makes them cross, but makes it impossible to complete a (F, F) input to a perfect H-matching (Figure 26). By reversing the inputs and outputs, we get a partial crossing mechanism that accepts everything but (T, T). We label the first one a true (T) partial crossing mechanism and the second a false (F) partial crossing mechanism.

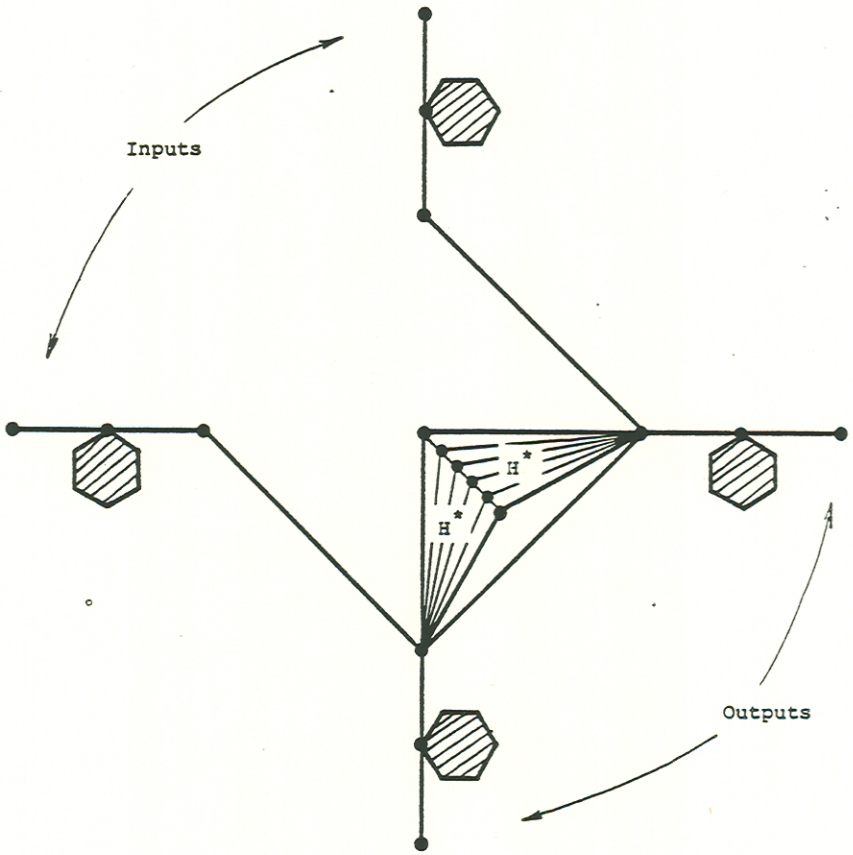

Figure 24: *A true (T) partial crossing mechanism for case 1.*

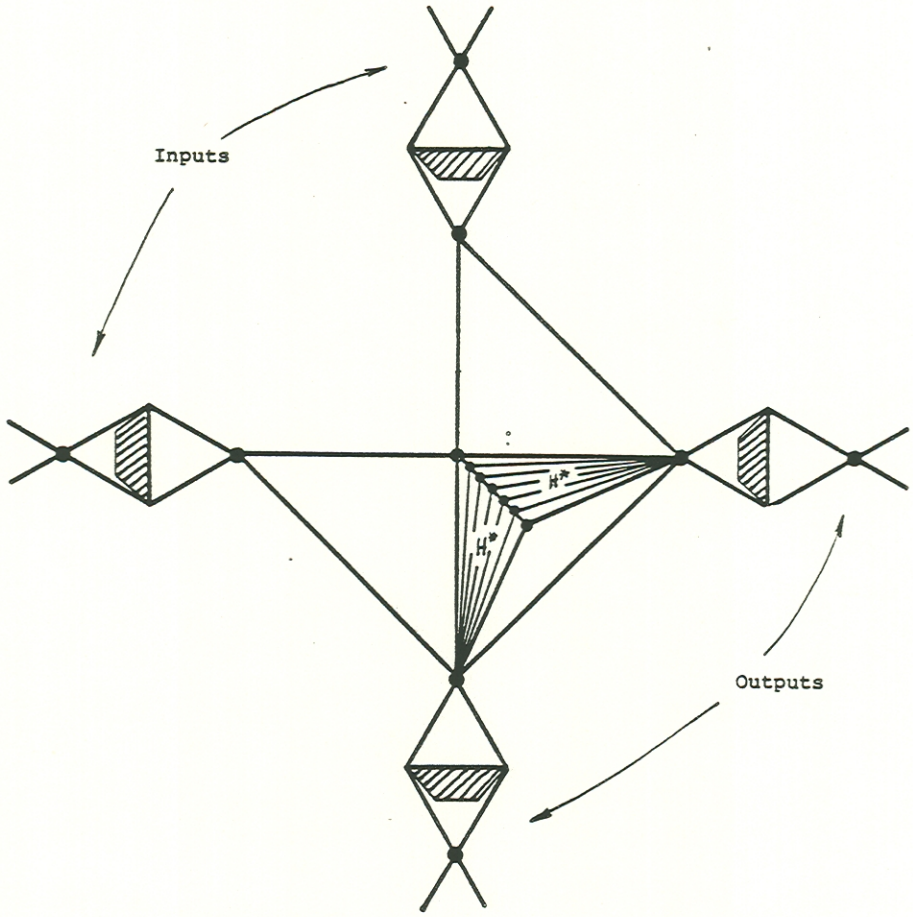

Figure 25: *A true (T) partial crossing mechanism for case 2.*

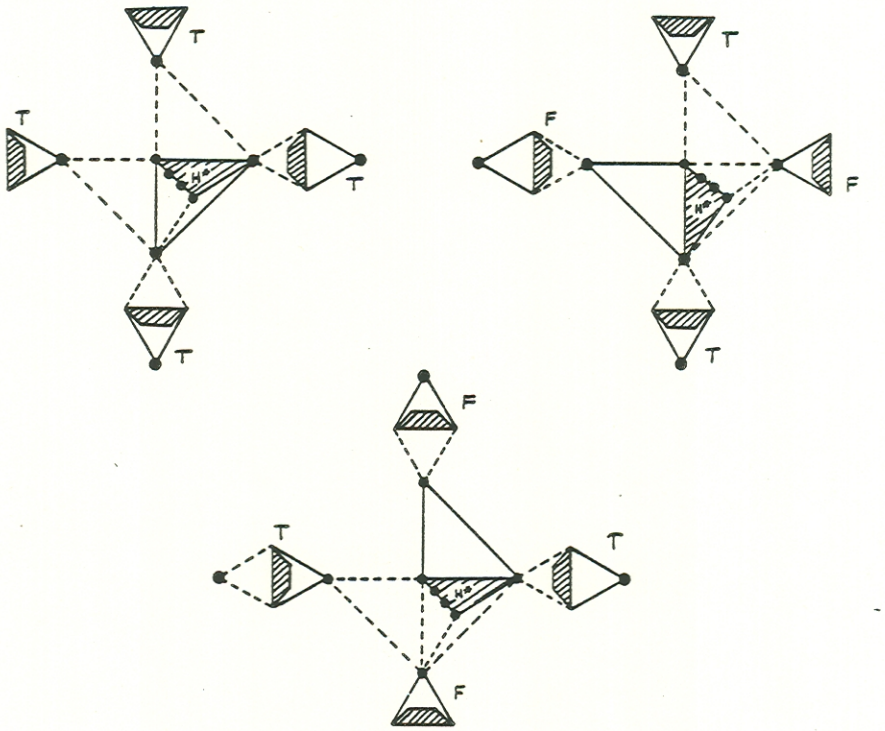

Figure 26: *Using the T partial crossing mechanism for case 2 to cross (T,T), (F,T) and (T,F) signals.*

Crossing Mechanism (Figure 27)

By combining branchings, inverters, and partial crossing mechanisms, we can construct a crossing mechanism as shown in Figure 27. If the left input is T, both branches are T. After the inverter, the lower horizontal lines are F, so all the partial crossing mechanisms on the vertical transmission line have at least one input of the proper kind. Thus these signals will cross. After the second inverter all horizontal signals are false, so the remaining partial crossing mechanisms work. A reversed branching with two false inputs gives a false output, so all inputs into the reversed inverter are F, giving an output of T.

If the left input is F, things get trickier. The branching must have one F and one T output. If the vertical input is false, then the top branch must be true. If the vertical input is true, the top branch must be the false branch. Otherwise not all the partial crossing mechanisms on the vertical line can be completed to a perfect matching. After the first two inverters, whichever case holds, one set of horizontal lines is F and the other is T. This means all of the partial crossing mechanisms on the right have one F and one T input, so they all work. Now, the reverse branchings all have one false and one true input, so all their outputs are T, and after the reversed inverter, we get an F signal, as we want.

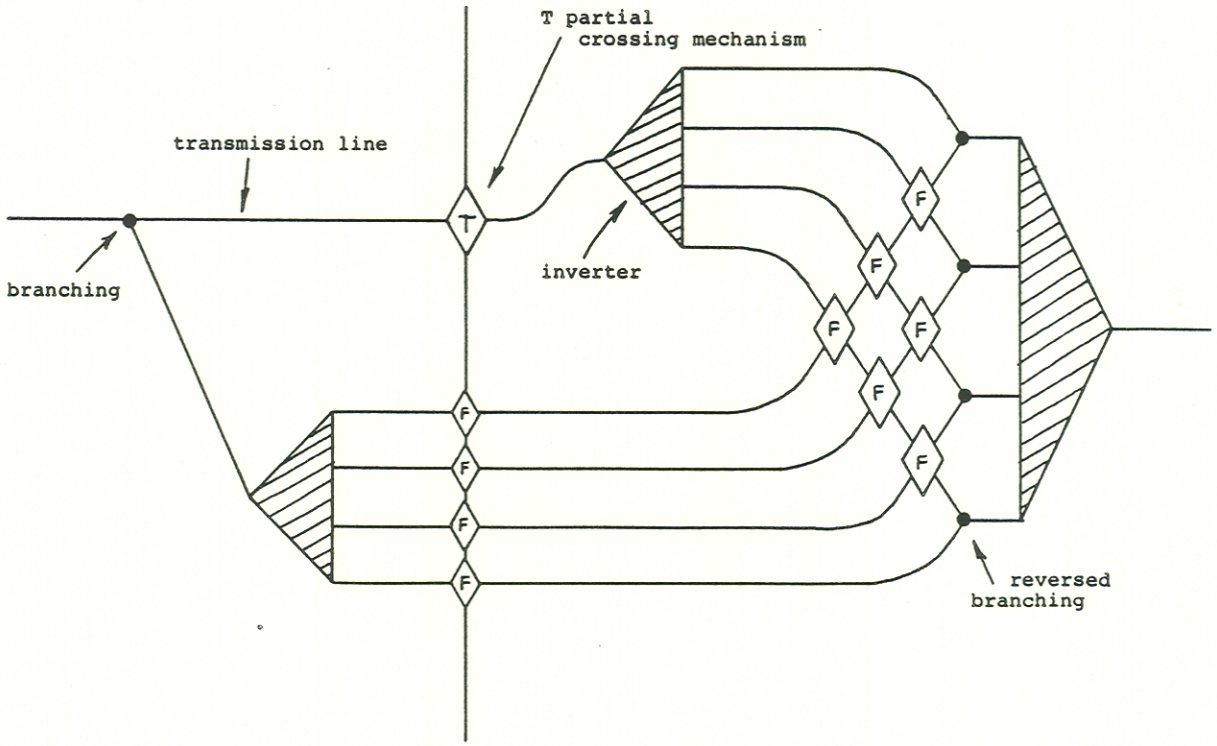

Figure 27: *A complete crossing mechanism.*

Loose Transmission Lines (Figure 28)

To tie up the leftover transmission lines so that all vertices can be used, first note that the number of vertices covered must always be a multiple of k. Thus, when we bring the loose lines together, the number of lines in false mode will always be the same, mod k. We add extra transmission lines in false mode (not connected to anything) to bring this to an exact multiple of k. Now, we notice that if we use an inverter with all k lines as input (Figure 28), which we will call an *ending*, then all k inputs must be the same in a perfect H-matching, but they can be either all true or all false. Thus, if we can bring the false lines together in k's, we can tie them off using this ending. Unfortunately, we don't know which lines are going to be false. However, we can still use this idea to end them.

To tie up the loose lines (assume there are exactly m of them), we put an $\binom{m-1}{k-1}$-fold branching on every line. We then construct $\binom{m}{k}$ endings and label each by a k-tuple of loose lines.

We then connect to each ending one of the outputs from the branching coming off the loose lines corresponding to the label given the ending. For example, see Figure 29. Note that we will need crossing mechanisms to do this.

Now, since the number of false loose endings is a multiple of k, we can partition them into disjoint k-tuples. By putting in false mode the transmission lines from the loose lines in a k-tuple to the ending labeled with that k-tuple, the other endings will have only true transmission lines leading into them. Thus, all the loose lines can be made part of a perfect matching. We now have a graph that has a perfect H-matching, if and only if the original 1-in-3 SAT problem is satisfiable.

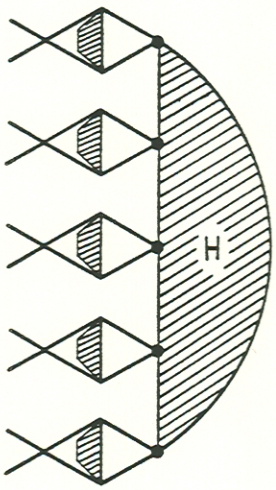

Figure 28: *An ending for case 2.*

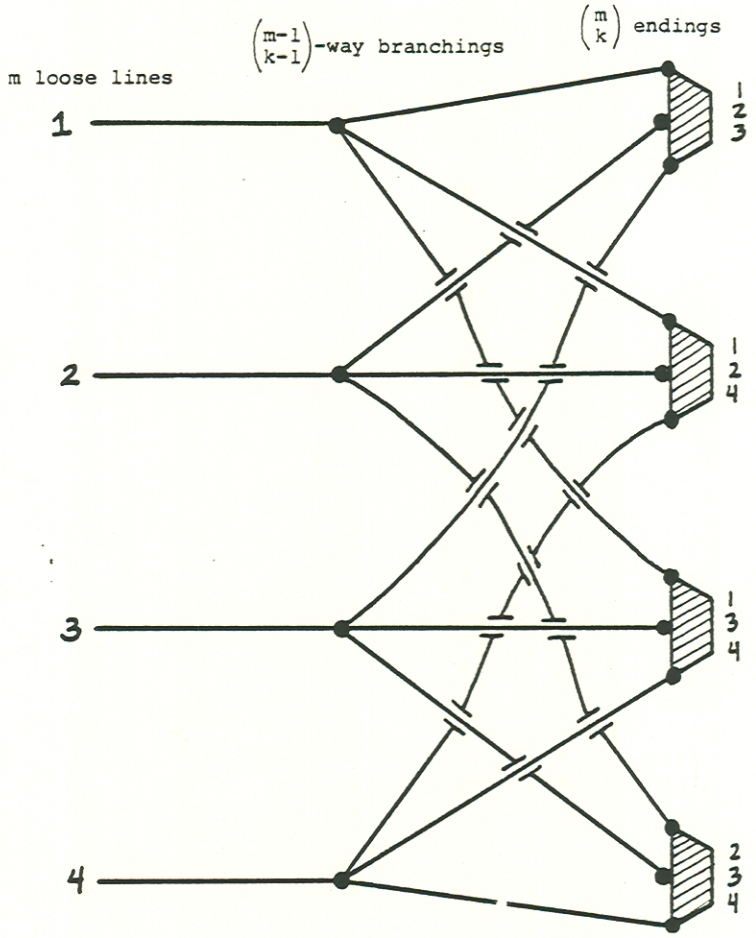

Figure 29: *Tying up the loose lines with endings.*

There is still one thing we must show, namely that no spurious copies of H can be put in our graph. This is easy to do for transmission lines, branchings, and inverters: by connectivity considerations, any other copy of H would disconnect the graph, leaving the wrong cardinality of vertices (i.e., not divisible by k) in one part. The only remaining building block (since endings are inverters, etc.) is the partial crossing mechanism. This is fairly easy to check also, using the fact that we picked a vertex of minimum degree in H when forming H^*. Since the number of vertices covered on the partial crossing mechanism must be a multiple of k, we can conclude from the preceding that precisely two of the four vertices connected to transmission lines must be covered by the copy of H on the partial crossing mechanism. By looking at the six possible ways of choosing these two vertices, one can easily complete the proof. ∎

3.2 Linear Time Algorithms

Theorem 5: *There is a linear time algorithm for perfect planar H-matching whenever H is a triangulated graph with four or more nodes.*

Proof: First we notice that any embedding of a triangulated planar graph is triangulated. If G is composed of copies of H, then there will be a copy of H on the "inside" (i.e. none of the interior faces of this H enclose nodes). Since H is triangulated, there is a triangle in G enclosing $|H|$ nodes, including the nodes on the triangle.

On the other hand, if any triangle encloses exactly $|H|$ nodes, it must be a copy of H. To show this we use the fact that any completely triangulated graph with four or more nodes is 3-connected. If any copy of H contains a point inside and a point outside the triangle, this implies that it must contain all three vertices of the triangle. This is a contradiction since removing this copy of H leaves less than $|H|$ nodes enclosed by the triangle disconnected from the rest of G. Thus, if G can be decomposed into copies of H, this triangle and the nodes enclosed by it must form one such copy.

We now remove this copy of H, and proceed recursively. Since we can check for isomorphism with H in constant time, this gives a polynomial algorithm. As a corollary, if such a decomposition exists, it is unique.

To make this algorithm run in linear time, first identify all the triangles in the graph. This can be done by repeatedly identifying all triangles containing a node of low degree, and then deleting the nodes of low degree. Since a constant portion of the nodes in any planar graph have degree at most 6, this task can easily be accomplished in linear time.

Next, we store the triangles on a tree, where each triangle's descendants are those triangles it contains. By examining the triangles containing a given node, one can assign a tree structure to them in time proportional to the degree of the node. We can combine these local tree structures to obtain the desired tree by using depth-first search.

Now traverse the tree in post-order (leaves first), and at each node do the following: If the triangle corresponding to the node contains less than $|H|$ vertices of G, remove the node from the tree. If it contains exactly $|H|$ vertices of G, check if it is a copy of H. If not, return "no decomposition." If it is a copy of H, remove this leaf from the tree and all the vertices in this copy of H from G. If the triangle contains more than $|H|$ nodes, return "no decomposition." Depth-first search takes linear time. In the second part, we spend constant time on each triangle (unless we find more than $|H|$ nodes in it, in which case we stop immediately). Thus, the algorithm takes linear time. ∎

4. Applications

The techniques described in this paper can be applied to solve a variety of related problems. The *optimal tile salvage problem* is one example. The problem can be described as follows: Consider an $\sqrt{N} \times \sqrt{N}$ region of the plane tiled with unit squares, some of which have been removed. The tiles which remain represent functional chips and the tiles which have been removed represent faulty chips on a wafer. The *optimal x, y tile salvage problem* is to find the maximum number of functional, non-overlapping $x \times y$ tiled rectangles. The orientation of the rectangles does not matter and we assume without loss of generality that $x \leq y$. For $x = 1$ and $y = 1$,

the problem is trivial. For $x = 1$ and $y = 2$, the problem can easily be solved as an instance of the usual maximum matching problem. On the other hand, Fowler, Paterson and Tanimoto [5] showed that the optimal tile salvage problem is NP-complete if $x = 3$ and $y = 3$. By applying the techniques developed in this paper, *all* but the trivial cases (1×1) and (1×2) are easily shown to be NP-complete. This is because the generator, receptor, and transmission line gadgets of Section 2 can also be modified to work in a grid setting for any rectangle with $x = 1$ and $y \geq 3$ or $x \geq 2$ and $y \geq 2$. For example, a generator for the 2×2 tiling problem is shown in Figure 30. Notice the close relationship between the tiling generator and the graph generator in Figure 6. A complete set of gadgets (e.g., receptors, transmission lines and generators) is included in Figures 31-33.

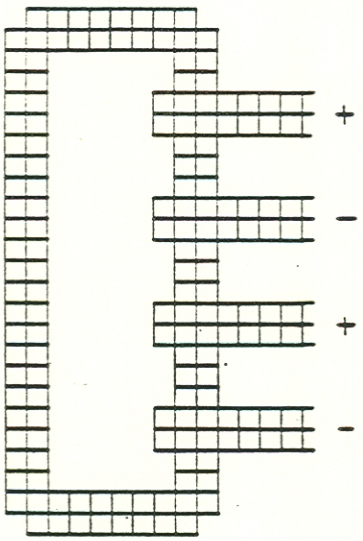

Figure 30: *A generator for the 2×2 optimal tile salvage problem. When in true mode, the leftmost and rightmost pairs of squares on the + lines can be used to form 2×2 rectangles, but not the leftmost and rightmost pairs of squares on the − lines. The reverse is true when the generator is in false mode.*

The approximation algorithm described in Section 2.2 can be very easily applied to grid problems since for most practical problems the cut made by the Lipton-Tarjan separator algorithm is likely to be a straight-line cut through the grid. The algorithm developed by Baker [1] also gives a very nice approximation algorithm for this problem.

It is likely that there are further applications of these techniques. For example, the gadgets seem to work for planar H-matching problems involving edge-disjoint graphs or induced graphs for many graphs H. The reduction can also be easily extended to give an alternate proof of the original Kirkpatrick–Hell [8] result for non-planar generalized matching, and for several other covering, packing and matching results [3-6].

As a final example, we apply a result of Johnson [7] to show that the "dots and boxes" game is NP-complete. In *dots and boxes*, two players take turns drawing unit length segments between

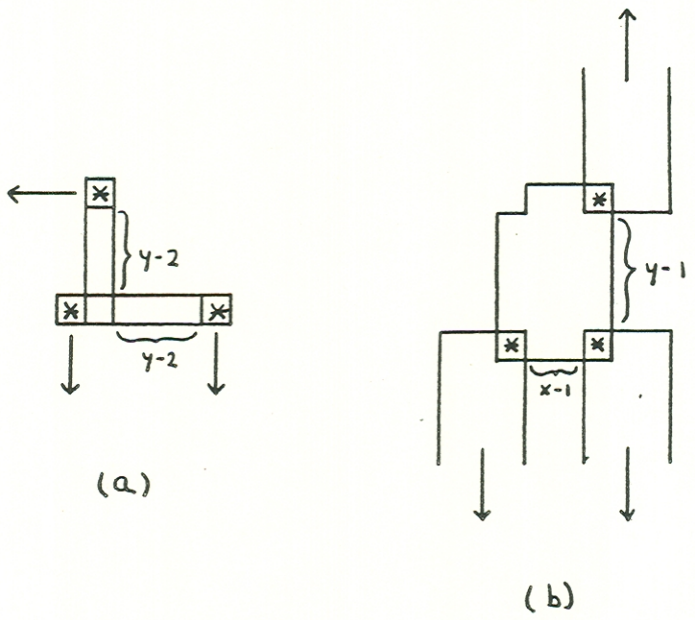

Figure 31: *Receptors for the $x \times y$ optimal tile salvage problem. The case when $x = 1$ and $y \geq 3$ is shown in (a), and the case when $x \geq 2$ and $y \geq x$ is shown in (b). Asterisks denote unit squares that serve as connection points to transmission lines. Arrows denote the origination and outgoing direction of transmission lines.*

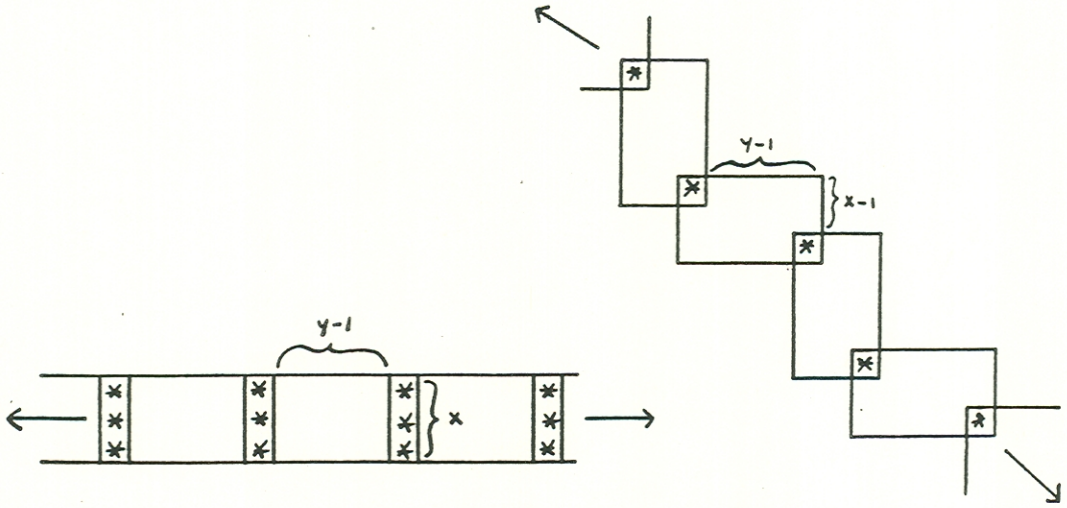

Figure 32: *Two types of transmission lines. Asterisks denote unit blocks connecting $x \times y$ rectangles.*

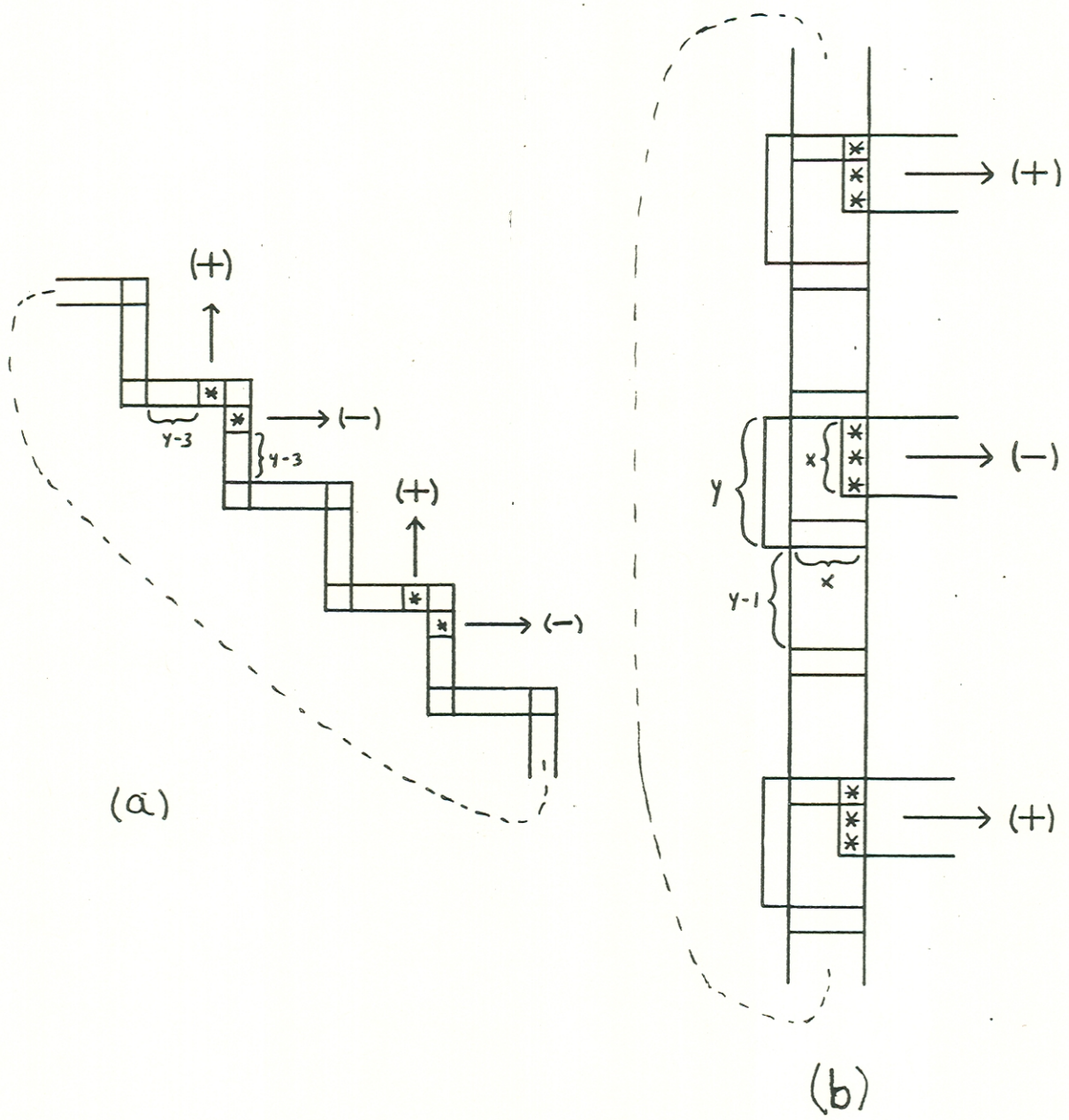

Figure 33: *Generators for the $x \times y$ optimal tile salvage problem. The case when $x = 1$ and $y \geq 3$ is shown in (a), and the case when $x \geq 2$ and $y \geq x$ is shown in (b). Asterisks denote unit squares that serve as connection points to transmission lines.*

consecutive points on an $N \times N$ grid. Whenever one player completes the perimeter around a unit square, he wins that square and draws another edge. The player winning the most squares wins the game. The problem is to decide whether or not a player has a winning strategy starting from a specified position (e.g., the input is a set of drawn segments and captured squares in the grid). In [7], Johnson shows that this problem is NP-complete if maximum K_3-matching is NP-complete for planar graphs with maximum node degree four. This is almost implied by the proof (in Section 2.1) that maximum planar K_3-matching is NP-complete. The only problem is that the receptor in Figure 2 could have degree six. In Figure 34, however, we illustrate an equivalent receptor with maximum degree four. Note that this receptor contains four node-disjoint triangles if and only if one of the connection nodes is not used by a generator triangle (e.g., if and only if the receptor is "true"). Otherwise, the receptor contains only three node-disjoint triangles. The remainder of the NP-completeness proof is identical to that in Section 2.1.

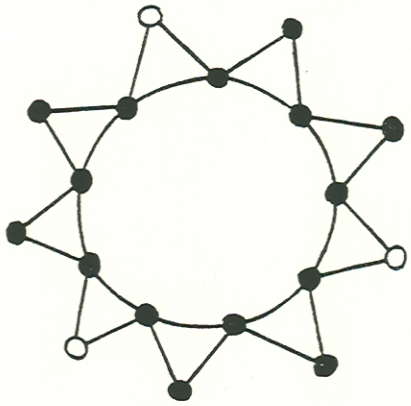

Figure 34: *Receptor for a 9-cycle with maximum node degree four. As in Figure 2, connection nodes are drawn as empty circles.*

5. Acknowledgements

We would like to thank Jon Buss, Dave Johnson, Ravi Kannan and Gary Miller for helpful discussions and comments.

6. References

[1] B. S. Baker, "Approximation Algorithms for NP-Complete Problems on Planar Graphs," *24th FOCS*, 1983.

[2] F. Berman, T. Leighton and L. Snyder, "Optimal Tile Salvage," Blue Chip Technical Report, Purdue University, 1983.

[3] M. E. Dyer and A. M. Frieze, "On the Complexity of Partitioning Graphs into Connected Subgraphs," CMU Graduate School of Industrial Administration Technical Report, 1983.

[4] M. E. Dyer and A. M. Frieze, "Planar 3DM is NP-Complete," unpublished manuscript, 1983.

[5] R. J. Fowler, M. S. Paterson and S. L. Tanimoto, "Optimal Packing and Covering in the Plane are NP-Complete," *Info. Proc. Let.*, 12(3), 1981, pp. 133-137.

[6] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, San Francisco, 1979.

[7] D. S. Johnson, "The NP-Completeness Column: An Ongoing Guide," *J. of Algorithms*, 4, 1983, pp. 397-411.

[8] D. G. Kirkpatrick and P. Hell, "On the Completeness of a Generalized Matching Problem," *10th STOC*, 1978.

[9] D. Lichtenstein, "Planar Formulae and Their Uses," *SIAM J. of Comp.*, 11(2), 1982, pp. 329-343.

[10] R. Lipton and R. Tarjan, "A Separator Theorem for Planar Graphs," *A Conf. of Theor. Comp. Sci.*, Univ. of Waterloo, 1977.