LA-UR- 98-2095

CONF-981123--

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE: **TOWARDS SYNTACTIC CHARACTERIZATIONS OF APPROXIMATION SCHEMES VIA PREDICATE AND GRAPH DECOMPOSITIONS**

AUTHOR(S): H. B. Hunt III, R. Jacob, M. V. Marathe, R.E. Stearns

SUBMITTED TO: IEEE AnnualSymposium on Foundations of Computer Science
San Francisco, CA
November, 1998

Towards Syntactic Characterizations of Approximation Schemes via Predicate and Graph Decompositions

HARRY B. HUNT III [1] RIKO JACOB [2] MADHAV V. MARATHE [2] RICHARD E. STEARNS [1]

Abstract

We present a *simple* extensible theoretical framework for devising polynomial time approximation schemes for problems represented using natural syntactic (algebraic) specifications endowed with natural graph theoretic restrictions on input instances. Direct application of our technique yields polynomial time approximation schemes for **all** the problems studied in [LT80, NC88, KM96, Ba83, DTS93, HM+94a, HM+94] as well as the *first* known approximation schemes for a number of additional combinatorial problems. One notable aspect our work is that it provides insights into the structure of the syntactic specifications and the corresponding algorithms considered in [KM96, HM+94]. The understanding allows us to extend the class of syntactic specifications for which generic approximation schemes can be developed. The results can be shown to be tight in many cases, i.e. natural extensions of the specifications can be shown to yield non-approximable problems. As specific examples of applicability of our techniques we get that

1. the problem of maximizing the number of satisfiable terms in a formula, where each term is represented explicitly by a bounded depth algebraic circuit with commutative and associative operators over a polynomially bounded domain and range, has PTAS when *restricted to planar instances*. Problems that can be naturally represented using this syntactic specification include maximization versions of constraint satisfaction problems in [Sc78, FV93, JCG97, KM96, HM+94] and graph problems considered in [Ba83, NC88, LT80, DTS93, HM+94a].

2. Simple extensions of our ideas can be applied to devise PTAS for the problem of maximizing (or minimizing) a linear objective function subject to linear packing (or covering) constraints when restricted to planar instances. Problems that can be represented using this specification include natural NP-hard packing and covering problems including those studied in [Sr95, PSW97].

Our results provide a non-trivial characterization of a class of problems having a PTAS and extend the earlier work on this topic by [KM96, HM+94].

[1]Email addresses: {hunt, res}@cs.albany.edu. Department of Computer Science, University at Albany - SUNY, Albany, NY 12222. Supported by NSF Grants CCR 89-03319 and CCR 94-06611.

[2]P.O. Box 1663, MS B265, Los Alamos National Laboratory, Los Alamos NM 87545. Email: jacob,marathe@lanl.gov. The work is supported by the Department of Energy under Contract W-7405-ENG-36.

1 Introduction and Motivation

In the past, extensive work has been done on the design of efficient approximation algorithms and schemes[3] for problems restricted to planar instances (See [Ba83, KM96, HM+94, CK95, NC88, LT80]). and the development of a theory of approximability based on the syntactic characterization of optimization problems. (See [KT94, PR93, KM+95, CK95, PY91] and related references) Recent results in [Ha97, RS97, AS97] show that in general, unless P = NP, a number of these problems are "hard" to approximate. Given these negative results, it is natural to consider restrictions on the general problem that are sufficient to ensure tractability (exact or approximate solvability). In this paper we concentrate on obtaining positive results and thus focus on the question raised by Khanna and Motwani [KM96] in this context: S*"Is there a syntactic characterization of* NP*-hard optimization problems having* PTAS*?"*

We make further progress in the direction suggested by the above remarks. Building on our earlier results and the results in Khanna and Motwani [HM+94, KM96], we present a simple extensible framework for devising generic approximation schemes for problems represented using natural syntactic specifications in which the input instances have a specified graph theoretic structure. Direct application of our ideas yield efficient approximation algorithms and approximation schemes for all the problems studied in [Ba83, KM96, HM+94, DTS93, NC88] and also for a number of additional important problems for which no previous results were known. These results significantly extend the a number of related results in [LT80, Ba83, Bo88, NC88, KM96, HM+94, HM+94a, DTS93, Ep95] and affirmatively answer recent open questions in [HM+94a, KM96, Ep95]. We describe the results in detail in Section 3. Our work is motivated in part by the following set of contrasting results for bounded versus unbounded arity predicates:

(1) For each fixed set of finite arity Boolean relation S, the problems MAX SAT(S) (see [HM+94, Cr95] for definitions) restricted to planar and near planar instances have a PTAS, [HM+94]

(2) the class of problems MPSAT informally defined as: Given a collection C of terms over n variables such that each term $\phi \in C$ is a *disjunction of* $O(n^{O(1)})$ *conjuncts*, find a truth assignment T maximizing the total number of the terms in C that are satisfied have a PTAS [KM96] and

(3) In a striking contrast to (2), a closely related class (obtained by simply interchanging the order of operators) informally defined as: Given a collection C of term over n variables such that each term $\phi \in C$ is a *conjunction of* $O(n^{O(1)})$ *disjuncts*, find a truth assignment T maximizing the total number of the satisfiable FOFs in C is readily seen to be NP-hard to approximate[4], even when restricted to planar instances.

Note in particular that while the result stated in (1) considers all *bounded arity set of relations*, results in (2) and (3) imply positive results only for *certain types unbounded arity relations*. Thus while in the case of bounded arity relations, the mere fact that the variable predicate bipartite graph is planar is sufficient to devise PTAS, the unbounded arity predicate case requires certain additional knowledge about the semantics of the predicates. The results and the proofs in this paper provide one possible explnation of these contrasting results. Specifically, our results are a step towards understanding class of unbounded arity predicates that are amenable to efficient approximations.

[3]Following [CK95, KM96], we define the class PTAS to consist of all NP-optimization problems having polynomial time approximation schemes.

[4]each ϕ can consists of a 3CNF formula

2 Preliminary Definitions

Given a set S of relations, where each $R_i \in$ S is specified by an explicit table, and an S-formula F, the problem MAX-RELATION(S) is to determine an assignment to the variables of F so as to maximize the number of terms satisfied. In this paper, we restrict our attention to variables $V = \{x_1, \ldots, x_n\}$ with domain $D = \{0, 1, , \ldots, poly(n)\}$; thus, we allow the domain size to grow polynomially with the number of variables in the formula. Although an adequate method of representing terms when they have fixed arity, the method of representing relations by tables can yield exponentially large when relations are non finite arity. The **bipartite graph** $BG(F)$ associated with the formula $F(P, V)$ defined as follows: The terms and variables in the formula F are in one to one correspondence with the vertices of the graph. There is an edge between a term node and a variable node iff the variable appears in the term. The **interaction graph** $IG(F)$ associated with the formula $F(P, V)$ defined as follows: The variables in the formula F are in one to one correspondence with the vertices of the graph. There is an edge $\{u, v\} \in E$ iff variables u and v appear together in some term of f.

The above representation of input instance (representing the functions in S and the graphical representation capturing the term-variable relationship) can be generalized in two independent directions -- term and graphical representation. We choose to represent each term as an algebraic circuit, in which the variables and the coefficients are allowed to take values from a polynomially bounded domain. For most part of this paper, we will also assume that the operators are commutative and associative multi-arity operators. The graphical structure associated with such a formula is a natural *extension* of the bipartite graph representation. The **circuit graph** $CG(F)$ associated with a formula $F(V, P)$ consists of one node for each variable, an algebraic circuit for each term and edge from a variable node x to an input node of a term circuit t labeled x denoting that x appears in t. The problem MAX-CIRCUIT-SAT is the following: Given as instance a circuit graph $CG(F)$ representing $F(V, P)$, find an assignment to the variables V to satisfy maximum number of terms in P. We use the phrase MAX-CIRCUIT-SAT restricted to planar instances to refer to the restriction when the graph $CG(F)$ is planar and the circuit correpsonding to *each term is of bounded depth.* Note that given a formula F there are a number of ways to construct $CG(F)$. In our input specification, we will assume that the gates are labeled with the operators and the associated semantics specified. In case of algebraic circuits, typically the semantics are well understood and therefore omitted. In general our sequence of transformation start with an instance $CG(F)$ and yield a new instance $CG(F')$ which can then be solved exactly in polynomial time.

Given $A \in [0,1]^{m \times n}$, $b \in [1, \infty)^n$ and $c \in [0,1]^n$ with $\max_j c_j = 1$, a packing (resp. covering) integer program PIP(resp. CIP) seeks to maximize (resp. minimize) $c^T \cdot x$ subject to $x \in Z_+^n$ and a system of linear constraints of the form $Ax \leq b$ (resp. $Ax \geq b$). Furthermore if $A \in \{0,1\}^{m \times n}$, we assume that each entry of b is integral. Consider the variant of PIP's (respectively CIP's) in which $A \in \{0, 1, \ldots poly(n)\}^{n \times m}$, $B = poly(n)$ and $x \in \{0, 1, \ldots poly(n)\}$. We call these Bounded packing (covering) programs and denote them by B-PIP and B-CIP respectively. We also use B-IP to denote bounded integer programs; i.e. programs of the form maximize $c^T \cdot x$ subject $A \in \{0, 1, \ldots poly(n)\}^{n \times m}$, $B = poly(n)$ and $x \in \{0, 1, \ldots poly(n)\}$ and $Ax = b$. Finally, define the variants B-IP(k) (respectively B-PIP(k) and B-CIP(k)) if each inequality contains exactly k terms. Given an instance I of a mathematical program with linear objective function, the graph $CG(I)$ is the same as the circuit graph associated with (P, V) where P is the set of constraints and V the set of variables. Thus for linear programs $CG(I)$ is identical to $BG(I)$.

We end with a few observations and remarks. Our definition of circuit graph associated with a formula is a departure from the definition of bipartite graph considered in [KM96]. Note that if $CG(F)$

is planar then $BG(F)$ is planar; the converse is not necessarily true. The graph $CG(F)$ can be seen as a refinement of the $BG(F)$ depicting the internal graphical structure of each term.

3 Summary of Results

The main contributions of this paper are three folds. First, we (i) identify *new* and *natural* syntactic languages (specifications) to specify problems and (ii) identify new graph theoretic restrictions on the underlying inputs that imply PTAS for problems so specified. Second, to obtain our results we propose two new theoretical concepts: (i) predicate decomposability and (ii) approximation (optimum) preserving reductions that also preserve graphical structure of the underlying instances. These concepts coupled with a new algorithmic technique referred as *structure preserving predicate decomposition* yields the necessary generic approximation schemes for problems represented using any of the syntactic specifications proposed here. Finally, we show that the circuit graph representation proposed above is in a sense more robust than the bipartite graph representation proposed in [KM96]. The arguments are based on the following observations: (i) the problems in [KM96] can be reduced to corresponding problems for our representation in an approximation preserving way, and thus is not a loss of generality, (ii) the positive results for bipartite graph representation in [KM96] can not be extended in general (unless P = NP) to more complicated predicates (e.g. nested formulas of depth more than 2), (iii) in contrast the results here based on the circuit graph lend themselves to immediate generalizations. The main theorem of this can be stated as

Theorem 3.1 *The problem* MAX-CIRCUIT-SAT *has a PTAS when restricted to planar instances.*

Moreover, we show that a number of important class of combinatorial as well as graph problems when restricted to planar graphs (and in general graphs obeying LT-property, see Appendix for formal definitions) can be reduced to appropriate instances of MAX-CIRCUIT-SAT in a approximation preserving as well as graph structure preserving way. Specifically, the reductions devised have two important properties: (i) they can be carried out in polynomial time and (ii) if Π is restricted to planar instances, then the instance of MAX-CIRCUIT-SAT obtained as a result of the reduction is also planar. Thus each of these problems have a PTAS, when restricted to instances obeying the LT-property. We all such reductions *structure preserving* L-reductions.

Thus, our results provide a syntactic (algebraic) specifications, whose closure under appropriate approximation preserving reductions define a characterization of problems that have PTAS. This represents a non-trivial characterization (subsuming the earlier characterizations) of class of problems having a PTAS. Examples of problems that can be solved using our framework include the following (for several of these results no previous approximation algorithms were known):

(1) Each of the graph theoretic, logical and combinatorial problems considered in Baker [Ba83], Khanna and Motwani [KM96], Hunt et al. [HM+94], Nizhiseki and Chiba [NC88], Lipton and Tarjan [LT80], and Diaz et al. [DTS93]. Note that general instances of the problems considered here (e.g. maximum independent set) are often very hard to approximate [Ha97].

(2) Planar versions of covering and packing programs in which both variables and coefficients take values from $[0, poly(n)]$, where n is the number of variables. This includes each of the problems considered in Peleg, Schechtman and Wool [PSW97] and Srinivasan [Sr95] and a number of packing/covering problems studied in [PST95]. Illustrative examples include: fault tolerant dominating set and hitting set.

(3) The optimization versions of the Boolean generalized CNF satisfiability problems studied in Schaefer [Sc78] and the optimization versions of a number of constraint satisfaction problems studied

in Feder and Vardi [FV93] and Jeavons et al. [JCG97], including H-matching for fixed H.

(4) A number of graph theoretic problems restricted to planar hypergraphs with unbounded arity hyperedges, and of unbounded degree. Examples include simple B–matching, independent set, vertex cover, etc. In general, these problems are NP-hard and can be approximated to within a factor of OPT^2/n [Sr95, AEL88]. (Here, OPT denotes the optimal value and n represents the number of nodes in the problem instance.)

(5) Our ideas are also applicable to a class of graph theoretic problems for which no previous approximation schemes were known even for planar graphs. Given a problem Π, define the problem D2-Π (distance 2-Π) as the problem of solving Π in the square[5] of the given graph G. Our results yield PTAS for a number of problems Π when G is restricted to be planar and of bounded degree (in some cases the restriction is not required). Note that if G^2 is δ-near planar then the result is immediate from the previous discussion. As an example, consider the problem D2-max independent set: Given a graph G, find a maximum cardinality subset of vertices, such that the pairwise distance between them is at least 2 (in terms of the number of edges). Our results show that when G is planar and of bounded degree, there is an PTAS to solve this problem. Other examples include the D2-min dominating set and D2-vertex cover.

Extensions and Generalizations. First, our results on covering and packing integer programs can be extended substantially in two orthogonal directions, namely allowing **non-linear constraints**, and to cases where some of the variables take rational values. For example a constraints of the form $c_1x^2yz + c_2xy^2w + \cdots + c_nxyz^3 = b$, where c_i and b are integers taking values from a polynomially bounded domain can be easily handled in our case. The only requirement we place is on the structure of the circuit graph associated with such a set. We view this extension as significant; this to our knowledge represents a non-trivial class *non-linear programs* that have efficient exact or approximate solutions. As the later sections will demonstrate our results are essentially tight in the sense that simple extensions of many of the classes yield problems that are provably non-approximable. In contrast, we can show that the extension is likely to fail for general linear integer programs whose bipartite graphs are planar. We do this by showing — (i) the problem of deciding the feasibility of IP instances I restricted to $BG(I)$ being a tree, or $IG(F)$ being a series parallel graph is NP-hard, and (ii) approximating the objective function for instances whose bipartite graphs are planar is NP-hard.

Second, our PTAS can be extended to two orthogonal graph classes.. The first class is more specific to layouts which are *close* to planar (bounded genus and weak level treewidth property). We show that (i) most of the problems, have a PTAS when restricted to instances satisfying the weak level treewidth property and (ii) several important and well known classes of graphs including planar, bounded genus graphs, (r,s)-civilized graphs and a subclass of k-ply neighborhood systems satisfy the LT-property. In contrast, we show that general k-ply neighborhood graphs as well as k-neighborhood graphs defined by Teng et al. [MT+97] *do not obey* the LT-property. The second class of instances (δ-near genus) we consider is obtained by extending the graph theoretic structure of planarity. Several of our results can be extended to instances that are δ-near genus or δ-near (r,s)-civilized. Thus, our results show that for a number of problems *both* the graph theoretic structure and the information about specific layouts can be used to devise good approximation algorithms.

Third, the techniques can be used to design polynomial time algorithms for the path and clustering problems considered in Eppstein [Ep95] (with running times essentially identical to those in [Ep95]) when restricted to graphs obeying the LT-property. Finally, our results also provide more efficient ex-

[5]Given a graph $G(V,E)$, the square graph $G^2(V,E')$ is obtained by adding an edge between two nodes x and y whenever there is a path of length at most 2 between x and y in G.

ponential algorithms for NP-hard problems restricted to problems whose underlying interaction graphs obey the LT-property. Specifically, our new results on Tables and the concept of predicate decomposability strongly extends the class of problems easily expressible as GSPs, solvable in deterministic time $2^{O(\sqrt{n})}$ including counting problems and many problems for graphs with unbounded arity vertices and hypergraphs with both unbounded arity vertices and hyperedges. For example, we get that problems such as independent set, dominating set, vertex cover, etc for unbounded arity hypergraphs with bounded treewidth have PTAS.

The rest of the paper consists of discussion of selected results. A few additional details are also given in the appendix. We refer the reader to [GJ79, CK95] for basic definitions in graph theory, computational complexity and combinatorial problems considered in this paper.

4 Overall Technique and Preliminary Results

The basic idea behind our algorithms is similar to the *shifting strategy* first used by [Ba83, HM85, Ho96] for obtaining polynomial time approximation schemes (PTASs) for problems restricted to planar and geometric instances. The overall schemata consists consists of the following basic steps:

(1) Decompose the given graph (instance) into vertex (edge) disjoint subsets such that an (near) optimal solution to the subgraph (sub instance) induced by each subset can be obtained in polynomial time. (This step exploits the fact that the underlying graph is decomposable.)

(2) Reduce (using D-reductions) each sub-instance to easily solvable sub-instance (uses predicate decomposability).

(3) Solve each transformed sub-instance optimally using known methods (such as Theorem 4.1) developed in [SH95, HM+94] (This step uses the theory of efficient solvability of algebraic problems restricted to instances of bounded treewidth developed in [HM+94, SH95].)

(4) Use the problem specification to combine the solutions to each of the sub parts to obtain a solution for the entire instance.

The schemta outlined above is similar in spirit to that used in [Ba83, HM85, HM+94, KM96]; although needs a number of new technical ideas at each step. The main technical contribution of the paper is to devise methods to accomplish Steps (2) and (3) above. It should be noted that the ordering of steps is crucial to the performance of the algorithm. It might be tempting to try and carry out the reduction on the planar instance directly rather than carrying out the reduction for each individual pieces. Such an attempt fails to work due to the special nature of reduction used which do not preserve approximation schemes but are sufficient to derive optimal solutions for the original problem. The proof of the following theorem appears in the Appendix.

Theorem 4.1 *Let* S *be a finite set of finite arity functions. an let $k > 0$ be fixed. Then the following statements hold: (i) For each fixed $k \geq 0$, the problem* WT-MAX-FUNCTION*(*S*) has an exact* NC*-algorithm when restrictedd to instances f such that $tw(BG(f)) \leq k$. (ii)* WT-MAX-FUNCTION*(*S*) has an* NC*-approximation scheme. when restricted to instances f such that $IG(f)$ is planar. (iii) The problem* B-IP(k)*, restricted to instances I such that $BG(I)$ is of bounded treewidth has a polynomial time algorithm. (iv) The problems* B-PIP(k) *(*B-CIP(k)*) restricted to instances I such that $BG(I)$ is planar have a* PTAS.

Definition 4.2 *Let Π and Π' be two optimization (maximization or minimization) problems. We say that Π D-reduces to Π' (denoted by $\Pi \leq_D \Pi'$) if there are two polynomial time computable functions f and g and constants $\alpha, \beta > 0$, such that for each instance I of Π f produces an instance $I' = f(I)$*

of Π' with the optimas $OPT(I)$ and $OPT(I')$ respectively and given any solution of I' with cost c', g produces a solution of I with cost c such that $|c - OPT(I)| = |c' - OPT(I')|$.

Notice crucially that the *reductions are not approximation preserving*. But the reductions allow us to compute the optimal value for I from an optimal value for I'. We will need this property in our proofs and thus we summarize it below:

Proposition 4.3 *Let P, Q, R be three optimization problems. Then the following holds: (i)* D*-reductions compose; i.e., If $P \leq_D Q$ and $Q \leq_D R$ then $P \leq_D R$. (ii) If $P \leq_D Q$ and Q has a polynomial time algorithm then P has a polynomial time algorithm. (iii) If $P \leq_L Q$ with $\beta = 1$ then $P \leq_D Q$.*

5 PTAS for MAX-CIRCUIT-SAT for planar instances

In view of the discussion in the previous section, we only need show how to find optimal solution for MAX-CIRCUIT-SAT when restricted to instances of bounded treewidth. We achieve this by optimally transforming an instance I of MAX-CIRCUIT-SAT to an instance I' of MAX-FUNCTION(S) in such a way that $TW(CG(I) = O(TW(IG(I'))$. For clarity of exposition, we prove our result in a series of steps.

Consider a term of the form $c = (x_1 \odot x_2 \cdots \odot x_p)$, where $\odot$ is a multi-arity operator that is commutative and associative. Consider a formula $F = \wedge_i c_i$, where each c_i is in the form above. In our proofs transforming F, we will work with a very special kind of tree decomposition which we call *Special Tree Decomposition*. Consider the tree decomposition $\mathcal{T}$ of $BG(F)$, and consider single clause c_i. $\mathcal{T}_{c_i} \subseteq \mathcal{T}$ denote the part of the tree that corresponds to c_i. By the definition of tree-decomposition it is easy to see that $\mathcal{T}_{c_i}$ forms a connected component and is therefore a tree. For each set X_k associated with $\mathcal{F}$, let $S_k \subseteq X_k$ which contains elements from the set $\{c_i, x_1, \ldots, x_p\}$. Without loss of generality we can assume that c_i-tree $\mathcal{T}_{c_i}$ has the following properties: (i) $\mathcal{T}_{c_i}$ is rooted and edges directed towards the root (inward arborecense), (ii) the sets S_k at each leaf node are of the form $\{c_i, x_j\}$, (iii) for all sets of the form $\{c_i, x_j\}$ there is leaf node v such that $S_v = \{c_i, x_j\}$, and (iv) $\mathcal{T}_{c_i}$ is Binary. With these assumptions, the notion of lowest common ancestor (LCA) of two nodes in $\mathcal{T}_{c_i}$ is well defined. Then the PROCEDURE TRANSFORM-TERM consisting of performing the following iterative procedure: (1) Choose a set of leaf nodes that cover all the pairs $\{c_i, x_i\}$, where $x_i \in c_i$.

(2) Mark all these leaf nodes as "unprocessed".

(3) Repeat the following procedure:

(3a) Choose an LCA pq of maximum depth and of two nodes q and p containing distinct variables x_q and x_p and marked "unprocessed".

(3b) Set $S_p = S_p - \{c_i\} \cup \{y_{pq}, x_q\}$. Similarly set $S_q = S_q - \{c_i\} \cup \{y_{pq}, x_p\}$. For all nodes t on the path from q to p, set $S_t = S_t - \{c_i\} \cup \{y_{pq}, x_p, x_q\}$. Finally, set $S_{pq} = S_{pq} - \{c_i\} \cup \{y_{pq}\}$. Here y_{pq} is a new distinct temporary variable. Also add the clause $y_{pq} \equiv x_p \odot x_q$.

(3c) Mark leaf nodes q and p as processed and proceed.

(4) Mark the node pq as "unprocessed".

Finally add a clause y_i where y_i is the last temporary variable introduced in the above procedure. The procedure can be seen to replace the original clause c_i by a set of clauses c_i'. We do this for each clauses introducing new set of variables for each clause. Denote the new formula obtained by conjunction of the the clause groups by F_1. The following lemma summarizes the properties of the transformation PROCEDURE TRANSFORM-TERM.

Lemma 5.1 *Given a c-tree $\mathcal{T}_{c_i}$* PROCEDURE TRANSFORM-TERM *outputs a tree decomposition $\mathcal{T}_{c'_i}$ (with more elements at each node) of $IG(c'_i)$ such that the for each set $X_k(\mathcal{T}_{c'_i}) \leq 4 + X_k(\mathcal{T}_{c_i})$. S' are isomorphic. Moreover* PROCEDURE TRANSFORM-TERM *is a* D*-reduction.*

Proof: The first part of Lemma follows since we replace an element in the set S_k by at most 4 new elements. To prove part (2) first note that c'_i is equivalent to c_i. This is true crucially since $\odot$ is a commutative and associative operator by assumption and thus we can combine the variables $x_1, \ldots x_p$ in any order — in particular in the order of their proximity in $\mathcal{T}_{c_i}$. The proof can now be completed by noting that

(a) Any truth-assignment $\mathbf{V}$ to the variables of c_i can be extended uniquely to a truth-assignment $\mathbf{W}$ to the variables of c'_i such that all clauses of c'_i, except possibly the single variable clause y^i are satisfied by $\mathbf{W}$, and $\mathbf{W}[y^i_{p-2}] = 1$ iff $\mathbf{V}[c_i] = 1$. Conversely, any such truth-assignment $\mathbf{W}$ can be restricted to a truth-assignment $\mathbf{V}$ to the variables of c_i such that $\mathbf{W}[y^i] = 1$ iff $\mathbf{V}[c_i] = 1$. ■

We call such terms c as *structure preserving decomposable predicates* since intuitively we can decompose a large multi-arity formula as a conjunction of bounded arity formulas in a way that (i) preserves the original tree decomposition and (ii) is a D-reduction. We omit a formal definition here due to lack of space. Now consider the formula F itself. Note crucially that it was the commutativity and associativity of $\odot$ that allowed us to prove the second part of Lemma 5.1. We can apply the above procedure of converting a single c to each of the terms in a sequential order. Let $F^m = \bigwedge_i c'_i$ denote the new formula obtained as result of this complete transformation. Also let $\mathcal{T}^1, \mathcal{T}^2, \ldots \mathcal{T}^m$ denote the sequence of tree decompostions that we get when we process the terms $c_1, c_2, \ldots c_m$ in that order. Similarly $\forall i, 1 \leq i \leq m$, let F^i ($F^1 = F$) denote the sequence of modified formulas obtained. We make the following important observation: Transforming the tree $\mathcal{T}_{c_i}$ to $\mathcal{T}_{c'_i}$ does not affect the other trees $\mathcal{T}_{c_j}$, i.e. only the element c_i gets deleted while working with the tree $\mathcal{T}_{c_j}$. This allows us to inductively maintain that for each set X^j_i (the set associated with node j in tree $\mathcal{T}_{c_i}$ $X^j_i \leq 4 + X^j_i$. Note importantly that a node X^j can be a part of at most $TW(\mathcal{T})$ decompositions. Combining these observations with Lemma 5.1 we can prove the following theorem (proof omitted).

Theorem 5.2 *For all i, $2 \leq i \leq n$, the following statements hold: (1) The transformation from F^i to F^{i+1} is a* D*-reduction, (2) If $TW(CG(F^1) \leq 4TW(IG(F^m))$, (3) F^{i+1} is satisfiable iff F^i is satisfiable, (4) F^m and $\mathcal{T}^m$ can be obtained in time $O(size(F))$.*

Completing the reduction. We are now ready to complete the description and the proof of correctness for the general case, namely when each term is a general nested formula with polynomial domain and polynomially bounded intermediate values. Consider an instance $F(P, V)$ of a formula such that $CG(F)$ is of bounded treewidth. As in the previous case we transform each $c \in P$ in an approximation as well as tree decomposition preserving way. Each gate of the circuit defines a subterm; we begin processing a subterm defined by a gate that is farthest from the output gate. This subterm is replaced by a new term using transformation as given above. We then successively move to terms defined by gates that are closer to the output gate. The only crucial observation that we need to make is the following: Consider a gate g (and hence a subterm). Then the set of nodes in the tree decomposition of $CG(F)$ containing g form a connected subtree.

Additional Remarks. First, note that as a direct corollary of the above discussion and Theorem 4.1, we get a polynomial time algorithm for exactly solving B-IP when restricted to bounded treewidth instances. In fact, it shows that even for *non-linear constraints* whose circuit graph is bounded treewidth can be efficiently solved. Second, the fact that the operators are commutative and associative was a *sufficient* condition for exact solvability. Any formula that can be transformed in a structure preserving way can be solved in this manner. Section 6 discusses one such formula. Third, the result points out the robustness of our syntactic specifications — the only internal structure used to guarantee efficient solvability was the graphical structure. This should be contrasted with the discussion in the introduction. Fourth, the reductions outlined here are in fact more general — they can be simultaneosly used to show the hardness (easiness) of decision, counting, uniqueness, parity and other related questions for the problems under study. Finally, noting that this transformation can also be carried out for instances having non-constant treewidth, we get $2^{O(\sqrt{n})}$ algorithms for several combinatorial problems restricted to planar and bounded genus graphs.

6 PTAS for MPSAT

Next, we consider the problems MPSATand outline anapproximation preserving reduction to an appropriate MAX-CIRCUIT-SAT problem. This result provides insights into the question and the discussion at the end of Section 1 (introduction). The overall idea behind the reduction is the following: Each clause ϕ is first replaced by a formula so that the resulting formula has bounded arity predicates. Next, we replace each such formula by a circuit so that the resulting reduction is a D-reduction. This allows us to the instances I in which $BG(I)$ is treewidth bounded. We begin by describing the PROCEDURE TRANSFORM-MPSAT. First, we create a new formula $F_1 = (V_1, P_1)$ as follows: $V_1 = X \cup Y$, X is in one-to-one correspondence with the variables in V. The variables in Y are in one to one correpsondence with the FOF's in $P = \{p_1, \ldots p_m\}$. Let $p_i = t_i^1 \vee \ldots \vee t_i^{nt_{p_i}}$. recall that each term is of the form $w_1 \wedge w_2 \wedge \ldots w_r$, where each w_i is a literal. Now for each FOF p_i, do the following: For each $t_i^j \in p_i$, we create a set of q_j^i predicates, $h_{(i,j)}^l$. Add the 3CNF clauses $h_{(i,j)}^l(v_{p_i}, x)$ where, $1 \leq l \leq q_j^i$ and $x \in VAR(t_i^j)$. $h_{(i,j)}^l(v_{p_i}, x)$ is true iff v_{p_i} is not i or x has the value which which makes the literal for $x \in t_i^j$ true. Let C_i denote the conjunction of clauses obtained by tranforming p_i. Finally let the clauses in $F_1 = \wedge_{i=1}^m C_i$.

Lemma 6.1 *Let F be a* MPSAT*formula and F_1 be the* 3CNF *formula obtained by tranforming R using* PROCEDURE TRANSFORM-MPSAT. *The the following holds.*
(1) F_1 is satisfiable iff F is satisfiable.
(2) PROCEDURE TRANSFORM-MPSAT *runs in time $O(size(F))$.*
(3) The reduction is planarity preserving; thus F_1 is planar if F is planar.
(4) If $BG(F)$ has treewidth $\mathcal{T}$, then the treewidth of $IG(F_1)$ has treewidth $O(\mathcal{T})$.
(5) If $BG(F)$ has treewidth k, then F can be decided in time $O(n^k)$.

Proof: Consider an assignment which satisfies F. For each variable in X, the assignment to the variable is the same as the corresponding variable in V of F. For each variable v_{p_i} corresponding to a clause $p_i \in P$, we set $v_p = j$, where j corresponds to one of the term $t_i^j \in p_i$ that is true. Conversly, a solution for F_1 becomes a solution for F by setting the variables in V the same as their value in X. This proves part (1) of the theorem.

The proof of part (2) follows by observing that the reduction consists of replacing each MINTERM of size $O(n)$ in a FOF by a CNF formula of size $O(n)$; thus the transformation takes time $O(size(F))$.

To prove part (3) and (4), observe that for each of our predicates $h_{(i,j)}^l(v_{p_i}, x)$, there is an edge (p, x) in the bipartite graph $BG(F)$, so that $BG(F)$ and $IG(F_1)$ are isomorphic.

To prove the last part of the theorem, note that all the variables excepting v_{p_i} are binary variables; v_{p_i} has its domain $\{0, \ldots, n\}$. We can simulate each v_{p_i} using $O(\log n)$ Binary variables; thus increasing the treewidth to $O(k \log n)$. Now it is clear that solving a SAT(S) problem when S is a set of finite arity Boolean relations takes time $O(2^r n)$ time for instances with interaction graphs of treewidth r. In our case this translates into a $2^{O(k \log n)} n$ time which is really $n^{O(k)}$ algorithm. ■

Next, we describe PROCEDURE SOLVE-TW-MPSAT: for solving MPSAT on instances of bounded treewidth. First, obtain a new formula $F_1(V_1, P_1)$ from $F(V, P)$ using PROCEDURE TRANSFORM-MPSAT. By Lemma 6.1 the treewidth of $BG(F_1)$ is no more than ck for some constant k. Second, convert $F_1(V_1, P_1)$ into a new formula $F_2(V_2, P_2)$ by replacing each term by an equivalent circuit. Theorem 6.2 show that this transformation is a D-reduction. Moreover the treewidth of $IG(F_2)$ is no more than $c_1 k$, for some constant c_1. Third, we solve $F_2(V_2, P_2)$ optimally using Theorem 4.1. Finally, map the assignment to the variables obtained in Step 6 to an optimal assignment to the variables in F.

Theorem 6.2 *Let F, F_1 and F_2 be obtained as above in* PROCEDURE SOLVE-TW-MPSAT. *Let $tw(F) = k$. Then the following holds.*

(1) $tw(F_2) = c_1 tw(F)$, *for some constant c_1, whenever $tw(F_1) \geq 4$.*

(2) The transformation from F to F_2 constitutes a D*-reduction.*

(3) PROCEDURE PTAS-MINSAT *runs in time $n^{O(k)}$. Thus the problems* MPSAT *can be solved optimally in time $n^{O(k)}$ for graphs of treewidth $O(k)$.*

Proof: Consider the tree decomposition $\mathcal{T}$ of $BG(F)$, and consider single FOF (term) $C_j = (m_1 \vee m_2 \ldots m_n)$, where crucially the $m_i's$ are the minterms. Let $\mathcal{F} \subseteq \mathcal{T}$ denote the part of the tree that corresponds to C. By the definition of tree-decomposition it is clear that $\mathcal{F}$ forms a connected component and is therefore a tree. Note that by Lemma 6.1, the tree decomposition of $BG(F_1)$ is almost the same, excepting that some of the sets associated with each node in the tree are slightly larger. We will work with the fragement associated with the term C in $BG(F_1)$ from now on. By following the same sequence of steps as outlined in Section 5 for MAX-CIRCUIT-SAT , we can replace the term C_j by the new description $C_j \equiv \bigwedge_{i,j,k} h_{j,k}(v_j, x_i)$, where $h_{j,k}$ are the new relations added and v_j is the auxillary variable corresponding to the term C_j. Note that since AND is a commutative and associative operator, we can form a new circuit that combines them in any order — in particular in the order of their occurrence in the decomposition tree. The Step of converting F_1 to F_2 is very similar to the PROCEDURE TRANSFORM-SAT in Section 5 and consists of replacing the set of clauses in C_j by a new clauses by interpreting C_j as a circuit. Example 8.1 depicts the construction. The details are straight forward. We only make one crucial observation. The circuit can combine two clauses one of the form $h_{j,p}(v_j, x_m)$ and $h_{j,q}(v_j, x_n)$; the only constraint placed on combining is their proximity in the C_j tree. This can be done since C_j is essentially flattened out and all the clauses are connected by AND. Such a rearrangement is **valid** because AND is a **commutative** and **associative** operator. It is now easy to see that the treewidth of $IG(F_2)$ is no more than a constant factor times the treewidth of $IG(F_1)$ and thus $IG(F)$. This completes the proof of Part (1). To verify Part (2) of the lemma, note that the new auxillary variables are functionally dependent on the old ones. The proof now follows along the same lines as the proof of Part (4) of Theorem 5.2. We now consider the running time of the algorithm. Note that v_j is a variable taking m_j distinct values, where m_j is the number of minterms in F. Thus the domain of each v_j is bounded by $O(n^\alpha)$, for some fixed $\alpha \geq 0$. As a result v_j can be represented by a $O(\log n)$ binary variables. This implies that the treewidth of F_2 as measured with respect to this new representation is $kO(\log n)$. The running time is thus $n^{O(k)}$. This completes the proof of the theorem. ■

Notes: (1) Intuitively speaking the clause $C = (m_1 \vee m_2 \ldots m_n)$, has been **flattened** to $\exists v{:}\, m_v$ which can now be used to write the clause as one big AND, allowing to mix the new atoms of the clause as needed. (2) Note that we reduced the problem for disjunction of minterms to the case of a SAT(S) problem where the relation set S is not fixed but grows with the instance. This is okay so long as it is easy (in our case constant time) to check if a given assignment to the variables is satisfiable. (3) The results demonstrate that the concept of circuit graph is sound and roboust in designing exact as well as approximation algorithms. (4) The results extend the results of Bodlaender [Bo88] and those of Stearns and Hunt [SH95] on certain types of predicates for which the resulting problems can be solved exactly. In particular it says that if the bipartite graph corresponding to a hypergraph *without a bound on arity of edges and degree of nodes* has bounded treewidth then several classical problems such as independent set etc has PTAS.

7 Packing and Covering Programs

Next, we discuss the extensions to finding PTAS for packing and covering programs when the corresponding restricted to planar instances. We first discuss the polynomial time solvability of the corresponding programs for instances of bounded treewidth. First note that by direct application of result in Section 5, the B-IP problem, restricted to bipartite graphs of bounded treewidth have a polynomial time algorithm. Thus the problems B-PIP and B-CIP restricted to bipartite graphs of bounded treewidth has a polynomial time algorithms. Next consider B-PIP (B-CIP) for planar instances. These problems have a PTAS and is proved by a direct application of arguments similar to the proof of Part (4) of Theorem 4.1. In contrast to the positive results in the earlier sections, we show that certain desriable extensions of these results fail.

Theorem 7.1 *(1) The* 0/1-IP*-feasibility problem is* NP*-hard even when restricted to instances I such that $BG(I)$ is a tree. The* NP*-hardness holds even for instances I with ($IG(I)$ with bounded treewidth; and variables ranging over $\mathbf{Z}^+$. (2) Unless* P = NP, *the problem* 0/1-IP *does not have polynomial time ϵ-approximable for any $\epsilon > 1$, even when restrcited to instances I whose $BG(I)$ is planar.*

Proof: Proof of part (1) is by a reduction is from the Partition problem. Let $S = \{a_1, \ldots a_n\}$ be an instance of the partition problem. Then checking if there is a subset $S_1 \subseteq S$ such that $\sum_{a_i \in S_1} a_i = B/2$ is equivalent to checking if the equation $\sum a_i x_i = B/2$ subject to the condition that $x_i \in \{0, 1\}$. It is easy to see that the bipartite graph is simply a star. To extend the result to hold for interaction graph having a bounded treewidth we introduce new variables y_i and then add then construct the following instance I: equations: $\forall 1 \leq i \leq n-2$, $a_i x_i + a_{i+1} x_{i+1} = y_i$ and $a_n x_n + y_{n-1} = B/2$. It is easy to see that that the treewidth of $IG(I)$ is bounded. The proof of part (2) follows from a reduction from Ex-1-3SAT. ∎

The hardness and the easiness results discussed above should be carefully compared. The hardness of 0/1-IP feasibility stems from the fact that we work with algebraic structures that are exponentially large (and in general infinite). Also, note that to obtain PTAS, we only needed the packing and covering constraints to satisfy the following requirement: The variables in the layers that are thrown away can be given values without destroying the feasibility of the constraints. This is similar to the *monotonicity* constraint considered by Feder and Vardi [FV93]. Packing and Covering programs by their nature provide sufficient conditions to achieve this.

Acknowledgments: It is a pleasure to acknowledge David Eppstein, Dorit Hochbaum, Viggo Kann, Sanjeev Khanna, Rajeev Motwani, Alessandro Panconesi, Ravi Kumar, S.S. Ravi, R. Ravi, Aravind Srinivasan and Sheng Hua Teng for helpful comments, criticism and pointers to related literature on subjects related to this paper. Much of the work done here was motivated by the questions and comments in [HM+94, Ep95, KM96, Sr95].

References

[JCG97] P. Jeavons, D. Cohen and M. Gyssens. Closure properties of constraints. *Journal of the ACM, (J. ACM)* 44(4):527-548, July 1997.

[AEL88] R. Aharoni, P. Erdős and N. Linial, "Optima of dual integer linear programs", *Combinatorica*, Vol. 8, 1988, pp. 13–20.

[AS97] S. Arora and M. Sudan, "Improved low-degree testing and its applications," *Proc. 29th Annual ACM Symposium on Theory of Computing (STOC)*, 485-496 (1997).

[Ar96] S. Arora, "Polynomial time approximation scheme for Euclidean TSP and other geometric problems," *Proc. 37th Annual Symposium on Foundations of Computer Science, (FOCS,* Burlington, pp. 14-16 October 1996.

[AG+98] S. Arora, M. Grigni, David Karger, P. Klein and A. Woloszyn "A Polynomial-Time Approximation Scheme for Weighted Planar Graph TSP," to appear in *Proc. SODA*, San Franciso 1998.

[BB72] U. Bertele and F. Brioschi, *Nonserial Dynamic Programming*, Academic Press, NY, 1972.

[Bo88] H. L. Bodlaender, "Dynamic programming on graphs of bounded treewidth," *Proc. 15th International Colloquium on Automata Languages and Programming (ICALP)*, LNCS Vol. 317, 1988, pp. 105-118.

[Ba83] B.S. Baker, "Approximation Algorithms for NP-Complete Problems on Planar Graphs," *Journal of the ACM (J. ACM)*, Vol. 41, No. 1, 1994, pp. 153-180.

[BKT96] I. Barland, P. G. Kolaitis and M.N. Thakur, "in teger Programming as a Framework for Optimization and Approximability," *IEEE Conference on Complexity Theory*, 1996, pp. 249-258.

[CK95] P. Crescenzi and V. Kann, "A Compendium of NP-Optimization Problems," January 1995.

[Cr95] N. Creignou, "A dichotomy theorem for maximum generalized satisfiability problems," *Journal of Computer and System Sciences*, 51(3):511-522, December 1995.

[DTS93] J. Diaz, J. Toran and M. Serna, "Parallel Approximation Schemes for Problems on Planar Graphs," *Proc. First European Symposium on Algorithms (ESA'93)*, Lecture Notes in Computer Science, Vol. 726, Sept. 1993, pp. 145-156.

[Ep95] D. Eppstein, "Subgraph Isomorphism in Planar Graphs and Related Problems," *6th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1995, pp. 632-640.

[Ep95a] D. Eppstein, Private Communication, 1995.

[FV93] T. Feder and M. Vardi, "Monotone Monadic SNP and Constraint Satisfaction," *Proceedings of 25th Annual ACM Symposium on the Theory of Computing (STOC)*, May 1993, pp 612-622.

[FW82] M. L. Fisher and L. A. Wolsey. On the greedy heuristic for continuous covering and packing problems. *SIAM J. on Algebraic and Discrete Methods*, 3:584–591, 1982.

[GJ79] M.R. Garey and D.S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, Freeman, San Francisco CA, 1979.

[Ha97] J. Hastad. Some optimal inapproximability results. *Proc. 29th Annual ACM Symposium on Theory of Computing*, pp. 1-10, El Paso, Texas, 4-6 May 1997.

[Ho96] D. S. Hochbaum (Editor), *Approximation Algorithms for NP-Hard Problems*, ¡to appear (PWS Publishing Company, Boston, MA, 1997).

[HM85] D. S. Hochbaum and W. Maass, "Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI," *J. ACM*, Vol 32,No. 1, 1985, pp. 130-136.

[HM+94] H.B. Hunt III, M.V. Marathe, V. Radhakrishnan, D.J. Rosenkrantz and R.E. Stearns, "Designing Approximation Schemes Using L-Reductions," in *Proc. of the 14th Annual Foundations of Software Technology and Theoretical Computer Science (FST&TCS)*, Madras, India, December, 1994, pp. 342-353. A complete version titled "Parallel approximation schemes for planar and near-planar satisfiability and graph problems,"*Submitted for publication*, December 1997.

[HM+94a] H.B. Hunt III, M.V. Marathe, V. Radhakrishnan, S.S. Ravi, D.J. Rosenkrantz, and R.E. Stearns, "A Unified approach to approximation schemes for NP- and PSPACE-hard problems for geometric graphs." *Proc. 2nd European Symp. on Algorithms (ESA '94)*, Springer Verlag, LNCS Vol. 855, J. VanLeuwen(ed.), Utrecht, the Netherlands, pp. 424-435, Sept 1994

[KT94] P. G. Kolaitis and M.N. Thakur, "Logical Definability of NP Optimization Problems," *Information and Computation*, No. 115, 1994, pp. 321-353.

[KPR93] P. Klien, S. Plotkin and S. Rao "Excluded Minors, Network Decomposition and Multicommodity Flows," *Proc. 25th Annual ACM Symposium on Theory of Computing, (STOC)*, pp. 682-690, San Deigo CA, May 1993.

[KM96] S. Khanna and R. Motwani "Towards a Syntactic Characterization of PTAS" *Proc. 28th Annual ACM Symposium on Theory of Computing, (STOC)*, pp. 329-337, Philadelphia, PA May 1996.

[KM+95] S. Khanna, R. Motwani, M. Sudan and U. Vazirani, "On Syntactic versus Computational views of Approximability," *Proc. 35th IEEE Symposium on Foundations of Computer Science (FOCS)*, 1996, pp. 819-830.

[LT80] R. J. Lipton and R. E. Tarjan, "Applications of a Planar Separator Theorem," *SIAM J. Computing*, Vol. 9, 1980, pp. 615-627.

[MT+97] G. L. Miller, S. H. Teng, W. Thurston and S. A. Vavasis, "Seperators for Sphere Packings and Nearest Neighbor Graphs," *J. ACM*, Vol. 44, No. 1, Jan. 1997, pp. 1–29.

[NC88] T. Nishizeki and N. Chiba, *Planar Graphs: Theory and Applications, Annals of Discrete Mathematics*, Vol. 32, Elsivier Science Publications, Amsterdam, 1988.

[PR93] A. Panconesi and D. Ranjan, "Quantifiers and Approximations," *Theoretical Computer Science*, 107, 1993, pp. 145-163.

[PY91] C. Papadimitriou and M. Yannakakis, "Optimization, Approximation and Complexity Classes," *Journal of Computer and System Sciences (JCSS)*, No. 43, 1991, pp. 425-440.

[PSW97] D. Peleg, G. Schechtman and A. Wool. Randomized approximation of bounded multicovering problems *Algorithmica*, 18(1):44-66, May 1997.

[PST95] S. Plotkin, D. Shmoys and E. Tardos Fast approximation algorithms for fractional packing and covering problems. *32nd IEEE Annual Sym. on Foundations of Computer Science*, pages 495-504, San Juan, Puerto Rico, 1-4 October 1991. A complete version of the paper appears in *Mathematics of Operations Research*, 20:257–301, 1995.

[RS86] N. Robertson and P.D. Seymour, "Graph Minors II, Algorithmic Aspects of Tree-Width," *J. of Alg.*, 7, pp. 309-322, 1986.

[Ro82] A. Rosenthal, "Dynamic Programming is Optimal for Nonserial Optimization Problems," *SIAM J. Computing*, Vol. 11, 1982, pp. 47-59.

[RS97] R. Raz and S. Safra, "A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP," *Proc. 29th Annual ACM Symposium on Theory of Computing*, 475-484 (1997).

[SH95] R.E. Stearns and H.B. Hunt, "An Algebraic Model for Combinatorial Problems" *SIAM Journal on Computing* 25 (April 1996), pp. 448-476

[Sr95] A. Srinivasan "Improved approximations of packing and covering problems," In *Proc. ACM Symposium on the Theory of Computing*, pages 268–276, 1995.

[Sc78] T. Schaefer, "The Complexity of Satisfiability Problems," *Proc. 10th ACM Symposium on Theory of Computing (STOC)*, 1978, pp. 216-226.

Appendix

8 Additional Definitions and Proofs

For a set D and a natural number m, the set of m-tuples of elements of D are denoted by D^m. A subset R of D^m is called a m-ary relation over D; m denotes the arity of R (a function is defined similarly). We will also use the map $f : D^m \rightarrow D$ in our discussion. A *term* is a string of the form "$R(x_1, \ldots, x_k)$" where x_i are variables or constants and R is a k-ary relation (or function) over D. (The tuples of R_i indicate the allowed values that the variables $\{x_1, \ldots, x_k\}$ can take.) A formula F is a pair (V, P) where V is a set of variables and P is a set of terms such that $V \supseteq VAR(P)$. The problem WT-MAX-FUNCTION(S) is to assign values to each x_i, $1 \leq i \leq n$, so as to maximize $\sum w_{i_r} f_{i_r}(X_{i_r})$.

Proof of Theorem 4.1: Parts (1) and (2): The proof follows by a direct extension of ideas in [HM+94]. **Part (3):** The proof is obtained in two stages. First, by extending the ideas in [SH95], it follows that the problem B-IP(k), restricted to instances I such that $IG(I)$ is of bounded treewidth has a polynomial time algorithm. The only point to note is that the variables can take values from $\{0, 1, \ldots poly(n)\}$ and can be represented by $\Theta(\log n)$ binary variables encoding the binary representation. Thus if $IG(I)$ has treewidth k, then the new treewidth is $O(\log n)k$, resulting in an overall running time of $2^{O(\log n)k} = n^{O(k)}$. To complete the proof note that if $BG(I)$ has bounded treewidth l, and the number of variables in each inequality is bounded by a fixed constant k, then, there is a constant $\rho > 0$ such that $tw(IG(I) \leq \rho kl \leq \rho\, k\, tw(BG(I)$.
Part (4): To prove this part, we follow the idea given in [KM96]. Specifically, we break the graph into collection of layers obtained as a result of BFS. Consider a set it is $S_j = \cup L_k$ such that L_k are layers whose indices k are congruent to $2r$ modulo $2(p+1)$. Here the levels $L_1 \ldots L_r$ are divided into groups $S_0, \ldots S_p$. The proof now follows by observing the following:

1. By a simple averaging argument, it is clear that there exists a set S_j for which $\sum_{x_k \in S_j} c_k x_k$ is smaller than $\frac{OPT}{p+1}$. Thus, we can afford to assign a value 0 to the variables in these layers and still get a near optimal solution.

2. Setting the variables in layers that were thrown out to 0, we do not alter the feasibility of the inequalities; thus the inequalities are still satisfied.

3. The problem in each of the smaller pieces (subgraphs) consists of finding an optimal solution to **B-IP** when restricted to instances that are treewidth bounded. This can be done using the exact result for treewidth bounded instances..

The proof now follows. ■

Example 8.1 *Let p_i be the FOF given by $(x\bar{y} + zy + w\bar{x})$. We will create three predicates h_1, h_2 and h_3. We also have three new variable v_i, for the predicate p_i. Now we have the clauses $C_i^1 \equiv h_1(v_i, x) \wedge h_1(v_i, y)$ & $C_i^2 \equiv h_2(v_i, z) \wedge h_2(v_i, y)$ & $C_i^3 \equiv h_3(v_i, w) \wedge h_3(v_i, x)$*

The FOF p_i is replaced by $C_i^1 \wedge C_i^2 \wedge C_i^3$. The intended meaning of the predicates $h_i, 1 \leq i \leq 3$ is as given above. We note that the interaction graph corresponding to this fragment p_i is identical to the bipartite graph corresponding to p_i. Let the C_i tree be as shown in Figure 1. Now consider obtaining the formula F_2 from F_1. The formula is given by:

$$(t_1 \equiv h_{i,1}(v_i, x)) \bigwedge (t_2 \equiv h_{i,3}(v_i, x)) \bigwedge (t_3 \equiv t_1 \wedge t_2) \bigwedge$$

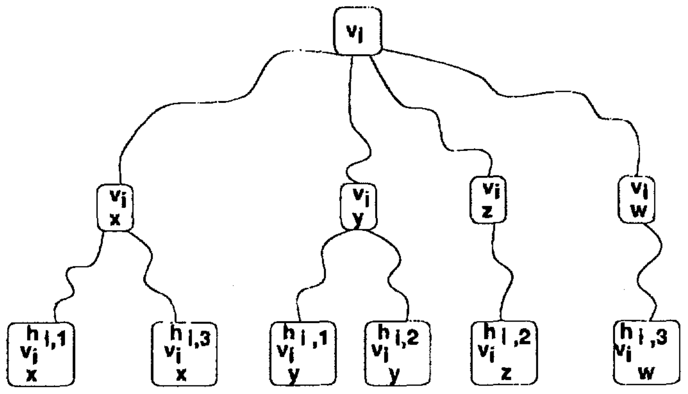

Figure 1: Figure shwing the C_i-tree corresponding to the bipartite graph for C_i in the formula F_1.

$$(t_4 \equiv h_{i,1}(v_i, y)) \bigwedge (t_5 \equiv h_{i,2}(v_i, y)) \bigwedge (t_6 \equiv t_4 \wedge t_5) \bigwedge$$
$$(t_7 \equiv h_{i,2}(v_i, z)) \bigwedge (t_8 \equiv h_{i,2}(v_i, w)) \bigwedge$$
$$(t_9 \equiv t_3 \wedge t_6) \bigwedge (t_{10} \equiv t_8 \wedge t_7) \bigwedge (t_{11} \equiv t_9 \wedge t_{10}) \bigwedge t_{11}$$

The tree decomposition for the bipartite graph associated with C_i as represented above can be easily constructed from the above formula and Figure 1.

9 Extension to Graphs obeying the LT-property

Next, we discuss the notion of *level-treewidth* property for a graph class $\mathcal{G}$. We first define the notion of level numbering – which is just an abstraction of the level numbers associated with a breadth first ordering of vertices.

Definition 9.1 *A* **level numbering** *of a graph $G(V, E)$ is a numbering of the vertices of the graph with the following properties:*

1. *Each vertex is assigned a unique number.*
2. *Letting $L_1, \ldots, L_p$ denote the set of vertices assigned levels $1, \ldots, p$, $\cup_i L_i = V$, $L_i \cap L_j = \phi$; $|L_i| \geq 1$ – thus implying that the level numbering partitions the set of vertices of G.*
3. *For all $1 \leq i \leq p$, if a vertex v assigned level i, then $N(v)$ – the neighbors of v are assigned levels $\{i-1, i, (i+1)\}$. Vertices at level 1 are have all their neighbours at levels 1 and 2 and similarly vertices at level p have all the neighbors at levels $(p-1)$ and p.*

The set L_i are sometimes referred to as levels; and we say that that level L_i is adjacent to level L_{i+1} and L_{i-1}, when these quantities are well defined. A subgraph induced by k consecutive levels L_r to L_{r+k} is the subgraph $\cup_{j=r}^{j=r+k} L_j$.

Definition 9.2 *A graph class G obeys the* level-treewidth *property (*LT-property*) if there is a polynomial time algorithm $\mathcal{A}$ that, for every $G \in \mathcal{G}$, assigns a level numbering to the vertices of G such that for all $k \geq 1$ the treewidth of the subgraph induced by k consecutive levels is $O(f(k))$.*

As in Eppstein [Ep95], a graph class $\mathcal{G}$ has a diameter-treewidth property (DT-property) if the treewidth of any graph in this family with diameter D is $f(D)$, for some function f.

In [Ep95], Eppstein characterizes those minor closed families that obey the DT-property. Extending his results, we obtain the second main theorem of this paper. The result implies the existence of PTAS for problems in the classes MPSAT, TMAX and TMIN when restricted to several well known classes of graphs. To prove this we only need to show that these graph classes obey the LT-property. Following [Ep95] define an *apex graph* G to be a graph such that for some vertex v (the apex) $G/\{v\}$ is planar. By using a key theorem proved by Eppstein [Ep95], we can show that:

Proposition 9.3 *Given an n-node graph G that is minor closed and and does not contain all apex graphs. Then G obeys the* LT-property.

Theorem 9.4 *Let G be a n-node (r,s)-civilized graph. The subgraph of G induced by the vertices in any k consecutive levels has treewidth $O(k)$. Thus the set of (r,s)-civilized graphs satisfy the* DT-property.

Proof: The algorithm $\mathcal{A}$ consists of dividing the plane into horizontal strips of width r and assigning level i to all the vertices that lie in strip i. Note that the vertices in k consecutive levels of a (r,s) civilized graph lie in a rectangular slice of side height $O(rk)$ and width $O(n)$. Since G is an (r,s) civilized graph, the maximum number of vertices in a rectangular region of dimensions $O(rk) \times O(s)$ is at most krs. Furthermore, removal of the vertices in this square breaks the graph into disjoint pieces. By recursively applying the above idea on each smaller piece, we can construct a tree decomposition of the graph G with treewidth $krs = O(k)$ since r and s are fixed. ■

A **neighborhood system** $\mathcal{N} = \{B_1, B_2, \ldots B_n\}$ is a finite collection of neighborhoods. For integers $k, d > 0$, we say that $\mathcal{N}$ is a k**-ply-neighborhood system in** d**-dimensions** if no point of $\Re^d$ is strictly interior to more than k of the balls. The **intersection graph** of a k-ply-neighborhood system is a graph in which each vertex corresponds to a neighborhood and there is an edge between two vertices iff the corresponding neighborhoods have a non-empty intersection. Intersection graphs of k-ply-neighborhood systems are a strict generalization of (r,s)-civilized graphs as well as planar graphs. The proof of the following theorem is omitted.

Theorem 9.5 *There exists family of k-neighborhood graphs and k-ply neighborhood graphs do not obey the level treewidth property.*

Define a **side-uniform** k**-ply neighborhood system** for **Rectangular neighborhoods**; each such neighborhood is a closed rectangle with fixed width but varying length. centered at p as follows: A **neighborhood system** $\mathcal{N} = \{B_1, B_2, \ldots B_n\}$ is a finite collection of neighborhoods. For integers $k, d > 0$, we say that $\mathcal{N}$ is a k**-ply-neighborhood system in** d**-dimensions** if no point of $\Re^d$ is strictly interior to more than k of the balls. The defintion of **intersection graph** of a side-uniform k-ply-neighborhood system is straightforward Notice that these graphs are a restriction of the earlier more general case. The advantage of these graphs though is the fact that they have nice decomposability properties as shown below.

Theorem 9.6 *The* **intersection graph** *of a side-uniform k-ply-neighborhood system obey the* LT-property. *In contrast there exists family of k-neighborhood graphs and k-ply neighborhood graphs do not obey the level treewidth property.*