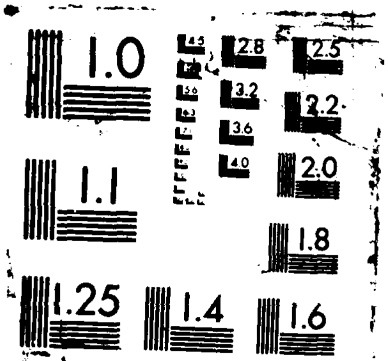
AD-A191 482 ALGORITHMS AND DATA STRUCTURES FOR AN EXPANDED FAMILY OF MATROID INTERSEC..(U) ROCHESTER UNIV NY DEPT OF COMPUTER SCIENCE G N FREDERICKSON ET AL. SEP 87 TR-227 1/1

UNCLASSIFIED N00014-86-K-0689 F/G 12/2 NL


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

(3)

AD-A191 482

DTIC FILE COPY

Algorithms and Data Structures for an Expanded Family of Matroid Intersection Problems

Greg N. Frederickson*
and
Mandayam A. Srinivas†

TR 227
September 1987

rochester
Department of Computer Science
University of Rochester
Rochester, New York 14627

3

Algorithms and Data Structures for an Expanded Family of Matroid Intersection Problems

Greg N. Frederickson*
and
Mandayam A. Srinivas†

TR 227
September 1987

*Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

†Department of Computer Science
University of Rochester
Rochester, NY 14627

---

*The research of the first author was supported in part by the National Science Foundation under Grant DCR-8320124 and by the Office of Naval Research under Contract N00014-86-K-0689. The research of the second author was supported in part by the Defense Advanced Research Projects Agency under Contract N00014-82-K-0193.

A191 482

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER TR 227	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Algorithms and Data Structures for an Expanded Family of Matroid Intersection Problems		5. TYPE OF REPORT & PERIOD COVERED technical report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Greg N. Frederickson and Mandayam A. Srinivas		8. CONTRACT OR GRANT NUMBER(s) N00014-86-K-0689 N00014-82-K-0193
9. PERFORMING ORGANIZATION NAME AND ADDRESS Dept. of Computer Science 734 Computer Studies Bldg. Univ. of Rochester, Rochester, NY 14627		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS DARPA / 1400 Wilson Blvd. Arlington, VA 22209		12. REPORT DATE September 1987 13. NUMBER OF PAGES 54
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Office of Naval Research Information Systems Arlington, VA 22217		15. SECURITY CLASS. (of this report) unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

data structures, degree-constrained spanning tree, matroid intersection, minimum spanning tree, on-line updating, partition matroid

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Consider a matroid of rank n in which each element has a real-valued cost and one of $d > 1$ colors. A class of matroid intersection problems is studied in which one of the matroids is a partition matroid that specifies that a base have q_j elements of color j, for $j = 1, 2, \ldots, d$. Relationships are characterized among the solutions to the family of problems generated when the vector $(q_1, q_2, \ldots, q_d)$ is allowed to range over all values that sum to n. A fast algorithm is given for solving such matroid intersection problems when d is

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT (Continued)

small. A characterization is presented for how the solution changes when one element changes in cost. Data structures are given for updating the solution on-line each time the cost of an arbitrary matroid element is modified. Efficient update algorithms are given for maintaining a color-constrained minimum spanning tree in either a general or a planar graph. An application of the techniques to finding a minimum spanning tree with several degree-constrained vertices is described.

$-A-$

Abstract. Consider a matroid of rank n in which each element has a real-valued cost and one of $d > 1$ colors. A class of matroid intersection problems is studied in which one of the matroids is a partition matroid that specifies that a base have q_j elements of color j, for $j = 1, 2, \cdots, d$. Relationships are characterized among the solutions to the family of problems generated when the vector $(q_1, q_2, \cdots, q_d)$ is allowed to range over all values that sum to n. A fast algorithm is given for solving such matroid intersection problems when d is small. A characterization is presented for how the solution changes when one element changes in cost. Data structures are given for updating the solution on-line each time the cost of an arbitrary matroid element is modified. Efficient update algorithms are given for maintaining a color-constrained minimum spanning tree in either a general or a planar graph. An application of the techniques to finding a minimum spanning tree with several degree-constrained vertices is described.

Keywords. data structures, degree-constrained spanning tree, matroid intersection, minimum spanning tree, on-line updating, partition matroid.

1. Introduction

Matroids are discrete mathematical structures that appear in a variety of applications. They are structures for which the greedy algorithm gives an optimal solution, and when intersected characterize such problems as minimum weight maximum cardinality bipartite matching ~~[L1]~~. In this paper we study a class of combinatorial problems from a matroid point of view. Consider a matroid in which each element has a real-valued cost, and one of d colors, for some constant $d > 1$. Given positive integers $q_1, q_2, \cdots, q_d$, we seek a base of the matroid that is of smallest cost subject to the constraint that it contain q_j elements of color j, for $j = 1, 2, \cdots, d$. For example, we can generalize the minimum spanning tree problem to a problem in which the edges have colors, and we desire a spanning tree of minimum cost subject to constraints on the number of edges of each color that are in the tree.

A *matroid* M consists of a set E of elements, and rules describing a property, called *independence*, of certain subsets of E. The rules satisfy axioms which may be found in [L1, W]. A maximal independent subset of E is called a *base*. A *matroid optimization problem* is the problem of finding a minimum cost base in a matroid in which a cost is associated with each element. For example, finding a minimum spanning tree of a connected graph is a matroid optimization problem, where the matroid consists of the set of edges in the graph, and independence corresponds to acyclicity. As stated above, matroid optimization problems can be solved by the greedy algorithm.

A *matroid intersection* involves two matroids defined on the same set E of elements, but with different sets of rules determining the independence of subsets in each matroid. A *matroid intersection problem* is an optimization problem whose solution is a subset of E of maximum cardinality that is independent in both matroids simultaneously, and is of minimum cost among all such subsets of E. While there is an algorithm for solving any given matroid intersection problem in polynomial time [L1, L2], the polynomial is large: at least $O(n^2m^2)$, where m is the number of elements, and n is the cardinality of the largest independent set. The special type of matroid intersection problem that we focus on in this paper is one in which each of the elements is labeled with one of d colors, and one of the matroids (a *partition* matroid) specifies that a certain number of elements of each color must be in the solution. In the case of $d = 2$ colors, the problem has been well studied, and more efficient solutions have been presented in [GT, G]. In this paper we explore the structure of d-color problems which allows for their efficient solution when $d > 2$.

The solution techniques of [GT, G] rely on finding a minimum cost solution from among only red elements and a minimum cost solution from among only green elements, and then pairing these red elements and green elements. However, for $d > 2$ colors, the analogue of such a pairing does not seem to exist. We overcome this difficulty by generalizing other characterization results in [GT, G]. We characterize the relationships among the solutions to a family of problems generated when the vector $(q_1, \cdots, q_d)$ is allowed to vary over all combinations which sum to n. The key relationship is the property of *dominance*, which allows us to search efficiently within the set of these $\Theta(n^d/d!)$ solutions. Dominance means that if one constrained minimum cost base dominates another with respect to the color constraints, then all elements of a certain color in the second base are in the first.

The dominance property makes possible a divide-and-conquer approach for finding a constrained minimum cost base that is efficient for small values of d. The algorithm runs in time in time $O(d\ T_0(m, n) + (d-1)!\ d!\ T(n, 2))$, where $T_0(m, n)$ is the time to solve an uncolored version of the problem, and $T(n, 2)$ is the time to solve the 2-color version given solutions for each color. For graphic matroids, it was shown in [FT, GGST] that $T_0(m, n)$ is slightly larger than proportional to m, and in [GT] it was shown that $T(n, 2)$ is $O(n \log n)$. The algorithm handles any d-color matroid intersection problem, such as scheduling unit-time jobs with release times and deadlines [GT], in essentially the same time bound. While the algorithm is factorial in d, it matches the bound in [GT] for $d = 2$ and is significantly more efficient than the previously known algorithms when d is a small constant.

We also address the problem of updating a solution repeatedly, as the cost of elements change one at a time. This on-line updating problem is a generalization of the 2-color update problem discussed in [FS]. We show how to use the dominance property to generate and maintain efficiently a sparse description of the $\Theta(n^d\ d!)$ solutions to all problems as the vector $(q_1, \cdots, q_d)$. We can update a d-color minimum spanning tree in $O(d^2 m^{1/2} + d^{8/3}(d!)^2 n^{1/3} \log n)$ time, and in $O(d^2(d!)^2 (\log d)^{-1/2}\ 2^{2\sqrt{2 \log(2d) \log n}} (\log n)^{3/2})$ time if the graph is planar. These match the update times in [FS] for the case when $d = 2$.

Our d-color algorithm can be used to find a multiple-degree-constrained spanning tree of a communications network. Suppose the degrees of a number d of the nodes are prespecified, because of the number of ports that they have. When $d = 1$, the problem is a special case of the 2-color minimum spanning tree problem [GT]. However, many interesting problem instances may require d degree-constrained nodes, where d is a small constant greater than one. We reduce this problem to a set of $(d-1)$-color problems, one of which yields the solution. While the problem is NP-hard for general d [GJ, p. 206], our algorithm is efficient for small d.

The remainder of the paper is organized as follows. In section 2 we introduce some terminology and new concepts that facilitate the later discussion. In section 3 we characterize the structure of d-color problem solutions, and establish the overall minimum cost, convexity and dominance properties. In section 4 we apply these characterizations to develop an efficient divide-and-conquer algorithm for the static d-color problem, and illustrate its efficiency for graphic matroids. In sections 5 and 6 we generalize the 2-color results of [FS] to d colors, and describe how to maintain a sparse description of certain arrangements of solutions to d-color problems to permit fast on-line update. In section 7 we discuss applications of our methods to other matroids and contexts.

2. Definitions

We identify some additional matroid terminology: a more complete discussion can be found in [L1, W]. The *rank* of a set $E' \subseteq E$, denoted as $rank(E')$, is the cardinality of a maximal independent subset of E'. Let B be a base, and f an element in $E-B$. The *circuit* $C(f, B)$ is the set consisting of every element that can be deleted from $B \cup \{f\}$ to restore independence. Let e be an element in B. The *cocircuit* $\overline{C}(e, B)$ is the set consisting of every element that restores rank to $B-\{e\}$. We will sometimes refer to an element in $C(f, B)$ as one *that f can replace in B*, and an element in $\overline{C}(e, B)$ as one *that can replace e in B*. Let M/E' denote the *contracted* matroid obtained from M by contracting the elements $E' \subseteq E$. The elements of M/E' are $E - E'$. If E' is independent, then the independent sets (bases) of M/E' are those sets $X \subseteq E - E'$ for which $X \cup E'$ is independent (a base) in M. We note that $rank(M/E') = rank(M) - rank(E')$.

For our problems on graphs, read *edge* for *element*, *spanning tree* for *base*, *cycle* for *circuit*, and *forest* for *independent set*. The *rank* is the number of edges in a spanning tree. Thus a minimum spanning tree is a minimum cost base of a graphic matroid. Similarly, for our unit-time job scheduling problem, read *job* for *element*, a *set of jobs with a feasible schedule* for an *independent set*, a *maximal such set of jobs* for a *base*, and a *minimal infeasible set of jobs* for a *circuit*. Thus a maximum-profit set of jobs with a feasible schedule is a maximum-cost base of a job scheduling matroid. Let $m = |E|$ and $n = rank(M)$.

We associate a *color* j, $j \in \{1, \cdots, d\}$ with each element in set E. For any set $E' \subseteq E$, let $colors(E')$ be a d-tuple $(i_1, i_2, \cdots, i_d)$ giving the count of elements of each color in E'. Let $c_0(e)$ be the positive, real-valued cost of element e, and $c_0(E')$ the total cost of elements in a set E'. For a given cost function, we refer to a base B in such a matroid as *a constrained minimum cost base*, or a *minimum cost base for its vector* $colors(B)$, if B is of minimum cost over all bases with the same *colors* vector. We assume that E has been augmented with elements of cost ∞ as necessary so that a base of each color $1, \cdots, d$ exists. Thus a *monochromatic minimum cost base* is a constrained minimum cost base whose *colors* vector has exactly one nonzero component.

Following [GT], we find it advantageous to extend the cost function so that each constrained minimum cost base B is unique for its vector $colors(B)$. We make two different extensions, both similar to extensions given in [GT]. We assume that a unique index is associated with each element. Let $\alpha = \min(\{|c_0(E') - c_0(E'')| : E', E''$ are sets of elements, $|E'| = |E''|$, $c_0(E') \neq c_0(E'')\} \cup \{c_0(e) : e$ in $E\})$. We define $c(e) = c_0(e) - \alpha / 3^i$, where i is the index of e. By our choice of α, we note that for any two distinct bases B_1 and B_2, $c(B_1) \neq c(B_2)$, and for any three distinct bases B_1, B_2 and B_3, $2c(B_2) \neq c(B_1) + c(B_3)$.

The second extension $c_L(\cdot)$ of $c_0(\cdot)$ is based on lexicography. Let $\bar{f} = (f_1(\cdot), f_2(\cdot), \cdots, f_d(\cdot))$ be a d-tuple of convex functions, and let π be any permutation on d-tuples. Let E' and E'' be sets of edges. We assume that $\bar{f}(colors(E'))$ yields d-tuple $(f_1(i_1), \cdots, f_d(i_d))$. Let $indices(E')$ be a sorted ordering of the indices of the elements in E'. Then we say that $c_L(E') < c_L(E'')$ if and only if one of the following holds, in which tuples are compared by lexicography.

1. $c_0(E') < c_0(E'')$
2. $c_0(E') = c_0(E'')$ and $\pi(\bar{f}(colors(E'))) < \pi(\bar{f}(colors(E'')))$
3. $c_0(E') = c_0(E'')$, $\pi(\bar{f}(colors(E'))) = \pi(\bar{f}(colors(E'')))$, and $indices(E') < indices(E'')$.

Note that for any two bases B_1 and B_2, $c_L(B_1) = c_L(B_2)$ implies that $B_1 = B_2$. It is clear that for any two bases B_1 and B_2 with identical *colors* vectors, and any $\bar{f}$ and π, $c(B_1) < c(B_2)$ if and only if $c_L(B_1) < c_L(B_2)$. Thus a constrained minimum cost base under $c(\cdot)$ is a constrained minimum cost base under $c_L(\cdot)$. We find $c(\cdot)$ more convenient in proving several key properties about d-color matroids, and $c_L(\cdot)$ more appropriate to use when designing algorithms for d-color matroids. When the cost function ensures that there is a unique base of minimum cost over all bases with *colors* vector $\bar{i}$, we call this base $B_{\bar{i}}$.

We next define the notion of a uniform cost adjustment with respect to each of the extended cost functions. The notion of a uniform cost adjustment comes from [G], where it was applied in handling 2-color matroids. A *uniform cost adjustment* with respect to $c(\cdot)$ consists of adding a constant δ_j to the cost of every element of color j in the matroid, for $j = 1, 2, \cdots, d$, and is specified by the d-tuple $\bar{\delta}$. A uniform cost adjustment with respect to $c_L(\cdot)$ consists of adjusting costs according to a d-tuple $\bar{\delta}$ and introducing a new d-tuple $\bar{f}$ of functions, along with permutation π. Since only differences in cost between elements of a particular color are significant in determining any constrained minimum cost base $B_{\bar{i}}$, the base $B_{\bar{i}}$ remains of minimum cost over the vector $\bar{i}$ after a uniform cost adjustment. Note that only differences in cost between various colors are significant in determining the relative costs of bases with different *colors* vectors. Furthermore, we can always assume without loss of generality that a uniform cost adjustment in a d-color matroid has at most $d-1$ nonzero components. The purpose of a uniform cost adjustment is to make some constrained minimum cost base $B_{\bar{i}}$ of overall minimum cost.

We say that a vector $\bar{i'}$ is a (j_1, j_2)*-neighbor* of $\bar{i} = (i_1, i_2, \cdots, i_d)$ if $i_{j_1}pr = i_{j_1} - 1$, $i_{j_2}pr = i_{j_2} + 1$, and $i_j' = i_j$ for all other j. Let the j_1*-negative neighbors* of $\bar{i}$ be the set of all (j_1, j_2)-neighbors of $\bar{i}$ with $j_2 \neq j_1$. Let the j_1*-positive neighbors* of $\bar{i}$ be the set of all (j_2, j_1)-neighbors of $\bar{i}$ with $j_2 \neq j_1$. When there is a unique minimum cost base for each vector $\bar{i}$, we extend the notion of neighbor from vectors to the bases that they index in the natural way. Let $\bar{i}$ and $\bar{i'}$ be the *colors* vectors of two bases. Suppose there is a *unique* color j for which $i_j > i_j'$. Then we say that $\bar{i}$ *dominates* $\bar{i'}$ *with respect to color* j, or that $\bar{i}$ j*-dominates* $\bar{i'}$.

Given a base B, a *swap* $s = (e, f)$ *available in* B is an ordered pair of elements, where $e \in B$, $f \notin B$, e and f are of different colors, and $C(f, B)$ contains e. Element f can be swapped in to replace element e, resulting in a base $B - \{e\} \cup \{f\}$ (denoted by $B + s$ or $B - e + f$). Given a base B, we say that a sequence S of swaps $s_1, \ldots, s_r$ is *available in* B if $B + s_1, \ldots, B + s_1 + \cdots + s_r$ are bases. Consider any cost function on E. Suppose swap sequence S is available in a constrained minimum cost base B. Let $s_i = (e_i, f_i)$ for $i = 1, \ldots, r$. We say that the sequence S is *optimal* if bases $B + s_1, \ldots, B + s_1 + \cdots + s_r$ are all constrained minimum cost bases. The sequence S is *color-conserving* if $colors(f_i) = colors(e_{i+1})$ for $i = 1, \ldots, r-1$. The sequence S is *acyclic* if $colors(e_i) \neq colors(e_j)$ for $i, j \in \{1, \ldots, d\}$ and $i \neq j$. Finally, the sequence S is *regular* if it is optimal, acyclic, and color-conserving. Note that any subsequence of a regular swap sequence is regular. We refer to a regular swap sequence S with $colors(e_1) = j_1$ and $colors(e_r) = j_2$ as a *regular* (j_1, j_2) *sequence*.

Let D be a set of bases with distinct *colors* vectors. The set D is *tight* if, for every pair of bases B_1 and B_2 in D, B_1 and B_2 are neighbors. A tight set D with $|D| = k > 1$ is *negative* if colors $j_1, \ldots, j_k$ can be uniquely assigned to bases in D such that for any base B in D, if base B is assigned color j, then every base in $D - \{B\}$ is a j-negative neighbor of B. A *positive* tight set is defined analogously, using j-postive neighbors instead of j-negative neighbors. If $|D| = 1$, then we arbitrarily assign the single base in D the color 1, and call D negative. We say that $hue(B)$ is the color assigned to B, and for any subset D' of D, $hue(D') = \bigcup_{B \in D'} hue(B)$. Let D be a negative tight set, B a base in D with $colors(B) = \bar{i}$, and $r = \sum_{j \in hue(D)} i_j$. Let $hspan(D)$ be the set of bases with *colors* vectors $\bar{i'}$ such that $\sum_{j \in hue(D)} i'_j = r$, and $i'_j = i_j$ for

$j \in hue(D)$. A tight set D is *complete* if $|D| = d$. We denote the unique complete, negative, tight set associated with a base B and color j by $D(B, j)$. Note that if $B, B' \in D(B, j)$ and B' is B's (j, l) neighbor, then $D(B, j) = D(B', l)$.

Let D be a negative, tight set of bases. The *swap graph* G_D associated with D has vertex set D and contains an edge (B_1, B_2) if and only if bases B_1 and B_2 are related by a single swap. If every constrained minimum cost base is unique for its *colors* vector, then there is a close relationship between negative tight sets of minimum cost bases and regular swap sequences. If D is a negative tight set of minimum cost bases and G_D is its swap graph, then every simple path in G_D corresponds to a regular swap sequence.

3. Characterization results

We first give several properties of 2-color matroids identified in [GT, G]. We then establish several important properties regarding constrained minimum cost bases and their neighbors, which hold for modified cost function $c(\cdot)$. First, there is a uniform cost adjustment that makes each constrained minimum cost base the overall (unconstrained) minimum cost base. Second, every pair of adjacent constrained minimum cost bases is related by a regular swap sequence of at most $d-1$ swaps. Third, if the *colors* vector of one minimum cost base dominates that of another with respect to a certain color, then all elements of that color in the dominated base are contained in the dominating base. Finally, we characterize how a constrained minimum cost base changes when the cost of one element changes.

Lemma 1 [GT, Thm. 3.1]. Consider a matroid with elements of two colors, red and green. Consider any positive, real-valued cost function. Let B_i be a constrained minimum cost base with i red elements. Executing a lowest cost red-green swap available in B_i transforms B_i into a constrained minimum cost base B_{i+1} with $i+1$ red elements. □

Lemma 2 [GT, Cor. 3.3]. Consider a matroid with elements of two colors, red and green. Consider any positive, real-valued cost function $c'(\cdot)$. Let B_{i-1}, B_i and B_{i+1} be constrained minimum cost bases with $i-1$, i and $i+1$ red elements, respectively. Then $c'(B_i) - c'(B_{i-1}) \le c'(B_{i+1}) - c'(B_i)$. □

The following result is implicitly stated in [G]. We supply an explicit proof, using Lemma 2.

Lemma 3. Consider a matroid with elements of two colors, red and green. Consider any positive, real-valued cost function $c'(\cdot)$. Let B_i be a constrained minimum cost base with i red elements. There exists a uniform cost adjustment for red elements that makes the cost of B_i less than or equal to the cost of every other cost base.

Proof. Let l be the smallest index such that B_l exists, and u the largest index such that B_u exists. It is observed in [GT] that B_i exists for each i, $l \le i \le u$. Assume as boundary conditions that $c'(B_{l-1}) = 2c'(B_l) - c'(B_u)$ and $c'(B_{u+1}) = 2c'(B_u) - c'(B_l)$. Take $\delta_{red} = c'(B_{i-1}) - c'(B_i)$. It follows from Lemma 2 by induction that $c'(B_{i'}) \ge c'(B_{i-1}) = c'(B_i) \le c'(B_{i''})$ for $l \le i' < i$ and $i < i'' \le u$. □

The following lemma, which is a variation of a lemma in [FS], establishes a fundamental property of bases in matroids.

Lemma 4. Let B be a base and e_1, e_2, f_1, f_2 be distinct matroid elements. Suppose $B - e_1 + f_1$ and $B - e_2 + f_2$ are bases, but $B - e_1 - e_2 + f_1 + f_2$ is not a base. Then both $B - e_1 + f_2$ and $B - e_2 + f_1$ are bases.

Proof: The proof is similar to that of Lemma 3 of [FS]. □

We next present some lemmas that will be useful in the proof of the overall minimum cost and dominance theorems for matroids with elements of $d > 2$ colors. Lemma 5 establishes that if an overall minimum cost property holds for constrained minimum cost bases, then the convexity property holds. Lemma 6 shows that if an overall minimum cost property holds for a certain subset of constrained minimum cost bases centered on a negative tight set, then a stronger version of an overall minimum cost property holds. Lemma 7 uses Lemma 6 to establish how the overall minimum cost property for a negative, tight set of constrained minimum cost bases impacts the connectedness of the corresponding swap graph. Finally, Lemma 8 uses the connectedness of the swap graph to establish the exact relationship between two neighboring constrained minimum cost bases for which the overall minimum cost property holds.

Lemma 5. Consider a matroid with elements of $d > 2$ colors. Let B_1, B_2 and B_3 be constrained minimum cost bases with respect to cost function $c(\cdot)$, such that B_2 is B_1's (j_1, j_2) neighbor and B_3 is B_2's (j_1, j_2) neighbor, for some j_1, j_2. Suppose each of B_1, B_2 and B_3 can be made an overall minimum cost base through some uniform cost adjustment. Then $c(B_2) - c(B_1) < c(B_3) - c(B_2)$.

Proof: Suppose in contradiction that $c(B_2) - c(B_1) \geq c(B_3) - c(B_2)$. Since B_1, B_2 and B_3 are distinct, this inequality must be strict, by definition of the modified cost function. Without loss of generality, suppose that B_1 is an overall minimum cost base. Let $\bar{\delta}$ be any cost adjustment vector that makes B_2 an overall minimum cost base. (By our initial assumption, $\bar{\delta}$ exists). Make all the adjustments of $\bar{\delta}$ except those for colors j_1 and j_2. Note that the new costs $c'(B_1)$, $c'(B_2)$, and $c'(B_3)$ have the same relative values as $c(B_1)$, $c(B_2)$, and $c(B_3)$. Now make the adjustments for colors j_1 and j_2, yielding costs $c''(B_1)$, $c''(B_2)$, and $c''(B_3)$. Since B_2 becomes an overall minimum cost base, we must have $c'(B_2) - c'(B_1) \le \delta_{j_1} - \delta_{j_2}$. We also get $c''(B_3) - c''(B_2) = c'(B_3) - c'(B_2) - (\delta_{j_1} - \delta_{j_2})$, which by the preceding argument is less than $c'(B_2) - c'(B_1) - (\delta_{j_1} - \delta_{j_2})$, which is at most $\delta_{j_1} - \delta_{j_2} - (\delta_{j_1} - \delta_{j_2}) = 0$. Thus $c''(B_3) < c''(B_2)$, which contradicts our assumption that a suitable δ exists. □

Note that Lemma 5 will hold for any cost function $c'(\cdot)$ derived from $c(\cdot)$ by a uniform cost adjustment.

Lemma 6. Consider a matroid with elements of $d > 2$ colors. Let D be a negative, tight set of constrained minimum cost bases for cost function $c(\cdot)$. Suppose for each base B in $hspan(D)$, there is a uniform cost adjustment that makes B an overall minimum cost base. Then there is a uniform cost adjustment that simultaneously makes every base in D of overall minimum cost, and every base in $hspan(D) - D$ not of overall minimum cost.

Proof : The proof is by induction on $p = |D|$. The basis case for $p = 1$ follows from our assumption that every base in $hspan(D)$, and therefore every base in D, can individually be made of overall minimum cost through a uniform cost adjustment. For the inductive step, assume $p > 1$. First perform a uniform cost adjustment to make some base B_1 in D, with hue j_1, of overall minimum cost. Let B_2 be a second base in D, with hue j_p. Consider the negative, tight set of bases $D_1 = D - \{B_2\}$, which is of size $p-1$. Since $hspan(D_1) \subset hspan(D)$, by the induction hypothesis we can perform a uniform cost adjustment such that every base in D_1, but no other base in $hspan(D_1)$, is of overall minimum cost. We next adjust the cost of color j_p so that the B_1 and B_2 are of the same cost. This does not affect which bases in $hspan(D_1)$ are of minimum cost among those in $hspan(D_1)$, since all bases in $hspan(D_1)$ have the same number of elements of color j_p. Since any two bases in D are of the same cost, by Lemma 5 the bases in D are the only bases in $hspan(D)$ of minimum cost within $hspan(D)$. Now make all colors in $hue(D)$ red, and the rest green. Note that one of the constrained minimum cost bases B_3 in this new problem is one of the bases of minimum cost in $hspan(D)$. By Lemma 3, there is a uniform cost adjustment that makes B_3 of overall minimum cost. This last adjustment will not alter the relative costs of any bases in $hspan(D)$, so that the bases in D will all be of the same cost, which will be an overall minimum. □

Lemma 7. Consider a matroid with elements of $d > 2$ colors. Let D be a complete negative tight set of constrained minimum cost bases with respect to $c(\cdot)$. Let D_1 be a negative tight subset of D such that every base in $hspan(D_1)$ can be made of overall minimum cost through a uniform cost adjustment, and every base in $D - D_1$ cannot be made of overall minimum cost by a uniform cost adjustment. Then the swap graph G_{D_1} is connected.

Proof: The proof is by induction on $|D_1| = p$. The basis case, in which $p = 1$, is seen to hold trivially. For the inductive hypothesis, assume that the lemma holds for any matroid with a complete negative tight set, and a negative tight subset of size less than p. For the inductive step, consider a matroid and sets D and D_1, with $|D_1| = p > 1$. Consider a connected component $D_2 \subseteq D_1$ of G_{D_1}.

We first argue that $|D_2| > 1$. Suppose $|D_2| = 1$. Let $B_1 \in D_2$, and without loss of generality assume that $hue(B_1) = green$. By Lemma 6, we can adjust costs uniformly so that B_1 is a base of overall minimum cost. Temporarily change every color other than green to red, so that the resulting matroid has only red and green elements. Note that B_1 is the minimum cost base for its *colors* vector. By Lemma 1, B_1 is related by a swap to some constrained minimum cost base B_2 with one fewer green element than B_1. If we restore the original element colors, it is apparent that B_2 is in $D_1 - \{B_1\}$, since these are the only green-negative minimum cost neighbors of B_1. By the definition of swap graphs, D_2 should then include B_2, a contradiction. Thus $|D_2| > 1$.

By Lemma 6, we can perform a uniform cost adjustment such that every base in D_2 is of overall minimum cost, and no other base in $hspan(D_2)$ is of overall minimum cost. We then change to green all colors in $hue(D_2)$. One of these bases, say B_1, will represent the component D_2 as a constrained minimum cost base in a matroid with $d - |D_2| + 1 < d$ colors. Clearly, $D_3 = D - D_2 \cup \{B_1\}$ is a complete negative tight set of bases in this new matroid, and $D_4 = D_1 - D_2 \cup \{B_1\}$ is a negative tight subset of D_3. Moreover, since $hspan(D_4) \subset hspan(D_2)$, every base in $hspan(D_4)$ can be made of overall minimum cost through some uniform cost adjustment, and since $D_3 - D_4 = D - D_1$, no base in $D_3 - D_4$ can be made of overall minimum cost. Note that two bases in the same connected component of G_{D_4} will be in the same connected component of G_{D_1}. By the inductive hypothesis, G_{D_4} is connected. Since the bases in $D_1 - D_2 \cup \{B_1\}$ are in the same connected component of G_{D_1}, and the bases of D_2 are in the same connected component of G_{D_2}, G_{D_2} is connected. $\square$

Lemma 8. Consider a matroid with elements of $d > 2$ colors. Let B_1 and B_2 be any two constrained minimum cost bases with respect to $c(\cdot)$ such that B_2 is B_1's j-negative neighbor, for some j. Let $B_2 \in D_1 \subseteq D(B_1, j)$. Suppose any base in $hspan(D_1)$ can individually be made of overall minimum cost through a uniform cost adjustment, and every base in $D(B_1, j) - D_1$ cannot be made of of overall minimum cost by a uniform cost adjustment. Then B_1 and B_2 are connected by a regular swap sequence of length at most $d-1$.

Proof : Since $D_1 \subseteq D(B_1, j)$, the swap graph G_{D_1} has at most d vertices. By Lemma 7, G_{D_1} is connected. Thus there is a simple path p of length at most $d-1$ between B_1 and B_2 in G_{D_1}. Let S be the corresponding swap sequence relating B_1 and B_2. Since p is acyclic and of length at most $d-1$, so is S. Since D_1 is tight and negative, S is color-conserving. Finally, since all bases in D_1 are constrained minimum cost bases, S is optimal. $\square$

We now establish the overall minimum cost and dominance properties.

Theorem 1. (Overall Minimum Cost) Let M be a matroid with elements of d colors, $d > 1$. Let B be a constrained minimum cost base with respect to cost function $c(\cdot)$. There exists a uniform cost adjustment that makes B of overall minimum cost.

Proof : The proof is by double induction, with the outer induction on d. The basis case, in which $d = 2$, follows from Lemma 3. For the inductive hypothesis, assume that the theorem is true for all matroids that have elements of at most $d-1$ colors. For the inductive step, consider a matroid of $d > 2$ colors. We prove the inductive step by induction on k, the number of elements of color 1. We will refer to color 1 as *green*.

For the basis, in which $k = 0$, we consider the original matroid with all green elements deleted. The basis case for k then follows from the inductive hypothesis for d. For the inner inductive hypothesis, assume that the theorem is true for all constrained minimum cost bases with at most $k-1$ green elements. For the inductive step, suppose $k > 0$.

Suppose the overall minimum cost property did not hold for some base B_1 with k green elements. We proceed to establish a contradiction. Consider the complete, negative, tight set $D(B_1,1)$ and the negative, tight set $D_1 = D(B_1,1) - \{B_1\}$. Every base in D_1 has $k-1$ green elements. By the inner inductive hypothesis, every base in $hspan(D_1)$ can be made of overall minimum cost. Thus by Lemma 6, we can adjust costs uniformly such that every base in D_1 is of identical, overall minimum cost in M, and no other base in $hspan(D_1)$ is of overall minimum cost. By temporarily changing every color other than green to red and applying Lemma 1, we conclude that for every base B in D_1 there is a base $mate(B)$ with k green elements such that B and $mate(B)$ are related by a swap. By Lemma 3, the cost of green elements can be uniformly adjusted, without disturbing the overall minimum cost property of any base in D_1, such that every base in $D_2 = \{mate(B) | B \in D_1\}$ is also of overall minimum cost. We have thus succeeded in uniformly adjusting costs such that every base in $D_1 \cup D_2$ is of identical, overall minimum cost.

Now consider any base B_2 in D_1. Suppose B_2 is B_1's (*green*, *red*) neighbor, and $mate(B_2)$ is B_2's (*blue*, *green*) neighbor. (Since, by our assumption, B_1 cannot be made of overall minimum cost and $mate(B_2)$ can, $B_1 = mate(B_2)$ and therefore $mate(B_2)$

cannot be a (*red*, *green*) neighbor of B_2). Let s_1 be the (*blue*, *green*) swap that transforms B_2 to *mate*(B_2). Since B_2 and *mate*(B_2) are of identical cost by our earlier cost adjustment, $c(s_1) = 0$.

We claim that swap s_1 is available in *any* base in D_1. In particular, s_1 is available in B_1's (*green*, *blue*) neighbor (and B_2's (*red*, *blue*) neighbor) B_3. This provides the desired contradiction: $B_3 + s_1$ has the same color combination as B_1 and the same cost as B_3, which is of overall minimum cost. Thus $c(B_1) \leq c(B_3)$, i.e., B_1 can be made of overall minimum cost through a uniform cost adjustment.

To prove the claim, we consider the regular (*red*, *blue*) swap sequence S_1 that, by Lemma 8, transforms B_2 into B_3. Let $|S_1| = p$. Note that every base in the sequence of bases induced by B_2 and S_1 is in D_1, and therefore every swap in S_1 is of zero cost. We establish by induction on p that s_1 remains available in a base B that is obtained from B_2 as a result of performing a sequence of p zero-cost swaps from a regular swap sequence.

The basis case for $p=0$ is trivial. For the inductive step, let $S_1 = S_2 s_2$, where S_2 is a regular (*red*, *purple*) swap sequence of length $p-1$ consisting of zero-cost swaps, and s_2 is a (*purple*, *blue*) zero-cost swap. By the inductive hypothesis, s_1 is available in $B_4 = B_2 + S_2$, which is in D_1. Now suppose s_1 is not available in $B_3 = B_4 + s_2$. Then, by Lemma 4, a (*blue*, *blue*) swap s_1' and a (*purple*, *green*) swap s_2' are available in B_4. Since $B_4 \in D_1$, it is of overall minimum cost. Therefore $c(s_1') \geq 0$. Since $c(s_1') + c(s_2') = c(s_1) + c(s_2) = 0$, $c(s_2') \leq 0$. Since $B_4 + s_2'$ has the same color combination as B_1, it follows that $c(B_1) \leq c(B_4 + s_2') \leq c(B_4)$, which is of overall minimum cost. By our assumption about B_1, this is impossible. Thus s_1 is available in B_3.

This completes the inductive step for k and the proof. $\square$

Theorem 2. (Dominance) Let M be a matroid with elements of d colors, $d > 1$. Let $B_{\bar{i}}$ and $B_{\bar{i}'}$ be constrained minimum cost bases with respect to $c(\cdot)$, such that $\bar{i}$ j-dominates $\bar{i}'$. Then every j-colored element in $B_{\bar{i}'}$ is in $B_{\bar{i}}$.

Proof: If $d = 2$, then the theorem follows from Lemma 1 and the fact that each constrained minimum cost base with respect to $c(\cdot)$ is unique for its *colors* index. If $d > 2$, we can construct a sequence of $k = i_j - i_j' + 1$ constrained minimum cost bases $B_{\bar{i}}, \cdots, B_{\bar{i}'}$, such that each base in the sequence is a j-negative neighbor of its predecessor. Consider any two bases B_1 and B_2 that are consecutive in this sequence, with B_2 the j-negative neighbor of B_1. By Theorem 1, every constrained minimum cost base can be made of overall minimum cost by a uniform cost adjustment. By Lemma 7, B_1 and B_2 are connected by a regular swap sequence S. Since S is regular, it is acyclic, which implies that every element of color j in B_2 is in B_1. The theorem then follows by induction on k. $\square$

To illustrate the properties of Theorems 1 and 2, we give an example of a graphic matroid. The edges of the graph will be of three different colors. Figure 1 gives the graph in terms of the three subgraphs of each color, red (solid lines), blue (dotted lines), and green (dashed lines). Each edge is labelled with its cost. In Figure 2 we list the solutions to all possible subproblems, each labeled with its cost. For example, the solution with one red, one blue, and two green edges is the third solution in the fourth row, and is labeled with the cost 16. We illustrate the overall minimum cost property by making base $B_{\bar{i}}$ be the unconstrained minimum-cost base over all bases, where $\bar{i}$ is for example

(1, 1, 2). This can be done if we add 6 to the cost of every blue element, and 4 to the cost of every red element. To illustrate dominance, consider the solutions for $\bar{i} = (0, 1, 3)$ and $\bar{i}' = (1, 2, 1)$. (We assume that red is color 1, blue is color 2, and green is color 3.) Here $j_{\bar{i}} = 3$, i.e., there are fewer green elements in $B_{\bar{i}'}$ than in $B_{\bar{i}}$, and at least as many elements of every other color. Thus the one green edge (of cost 4) in $B_{\bar{i}'}$ is in $B_{\bar{i}}$.

We next examine the impact of changing the cost of a single matroid element on a constrained minimum cost base. We begin as before with an earlier 2-color result, and proceed to generalize the result to $d > 2$ colors using the characterizations just developed.

Lemma 9 [FS, Thm. 2]. Let M be a matroid of red and green elements, with costs extended lexicographically to break ties. Let B_{i-1}, B_i and B_{i+1} be the constrained minimum cost bases with $i-1$, i and $i+1$ red elements, respectively. If one element in M changes cost, then B_i', the new minimum cost base with i red elements, will result from either B_{i-1}, B_i or B_{i+1}, with at most one element replaced in the appropriate base. Specifically, if a red element r_t increases in cost, then B_i' is the minimum cost base among the following three bases:

0. (or 3). B_i.

1. $B_i - r_t + r_a$, where r_a is the smallest cost red element that can replace r_t in B_i.

2. $B_{i+1} - r_t + g_a$, where g_a is the smallest cost green element that can replace r_t in B_{i+1}.

If a red element r_t decreases in cost, then B_i' is the minimum cost base among the following three bases:

0. (or 3). $B_{\bar{i}}$.

1. $B_{\bar{i}}-r_a+r_t$, where r_a is the greatest cost red element that r_t can replace in $B_{\bar{i}}$.

2. $B_{\bar{i}-1}-g_a+r_t$, where g_a is the greatest cost green element that r_t can replace in $B_{\bar{i}-1}$.

The cases for a green element changing in cost are analogous. $\Box$

We now give the generalization of the above result from 2 colors to d colors.

Theorem 3. Let M be a matroid with elements of d colors, $d > 1$. Let $B_{\bar{i}}$ be a constrained minimum cost base with respect to cost function $c(\cdot)$. If one element in M changes cost, then the new minimum-cost base $B_{\bar{i}}'$ will result from either $B_{\bar{i}}$ or one of its neighbors, with at most one element replaced in the appropriate base. Specifically, if a basic (nonbasic) element e (f) of color j_1 increases (decreases) in cost, then one of the following cases holds:

0. The new base $B_{\bar{i}}' = B_{\bar{i}}$.

1. $B_{\bar{i}}' = B_{\bar{i}} - e + f$, where e, f both have color j_1 and f (e) is the least (greatest) cost element of color j_1 that can replace e (be replaced by f) in $B_{\bar{i}}$.

2. There is a color j_2 such that $B_{\bar{i}}' = B_{\bar{i}'} - e + f$, where $\bar{i}'$ is a (j_1, j_2)-neighbor of $\bar{i}$ and f (e) is the least (greatest) cost element of color j_2 that can replace e (be replaced by f) in $B_{\bar{i}'}$.

Proof. We first consider the case where a basic element e of color j_1 increases in cost.

By Theorem 1 we can make $B_{\bar{q}}$ the unconstrained minimum-cost base, and therefore also the minimum-cost base over all bases with exactly i_{j_1} elements of color j_1, by uniformly adjusting the costs of all elements of colors $j \neq j_1$. Temporarily change the color of all j_1-colored elements to red and all other elements to green, so that $B_{\bar{q}}$ corresponds to red-green base $B_{i_{j_1}}$. We can then apply Lemma 9 with e in the role of r_t. If case 0 or 1 of Lemma 9 holds, then the corresponding case of our theorem holds. If case 2 of Lemma 9 holds, then there is a red-green base $B_{i_{j_1}-1}$ that differs from $B_{i_{j_1}}'$ by one element g_a. Let f be the element corresponding to g_a in the original matroid, and let $j_2 = color(f)$. Since g_a is the least cost replacement element over all green elements, f is certainly the least cost replacement element of color j_2.

The symmetric case of a nonbasic element f decreasing in cost is handled similarly. $\square$

Note that Theorems 1, 2 and 3 hold if cost function $c_L(\cdot)$ replaces cost function $c(\cdot)$ in the statement of the theorem. The use of $c_L(\cdot)$ has the advantage that arbitrarily many updates can be performed. This is not true for $c(\cdot)$, since changing the cost of one element can affect the value of α, which will alter the cost of every element.

4. Efficient solution of the static problem

We show how to find the constrained, lexicographically minimum cost base $B_{\bar{q}}$ consisting of q_j elements of color j, for $j = 1, 2, \cdots, d$, along with a uniform cost adjustment vector $\bar{\delta}$ that makes $B_{\bar{q}}$ of overall, unconstrained minimum cost. For matroids in which the contraction operation is reasonably efficient, the time to do this will be $O(d\,T_0(m,n) + (d-1)!\,d!\,T(n,2))$, where $T_0(m,n)$ is the time to solve an uncolored, or monochromatic, problem, and $T(n,2)$ is the time to solve a 2-color problem, given the constrained minimum cost bases for each color. Our algorithm first augments the set of elements with elements of large cost as necessary so that there is a base of each color, and finds monochromatic minimum cost bases for each color. This step accounts for the first term of the running time expression. The algorithm then calls a recursive routine to find the desired base and associated vector $\bar{\delta}$. This step accounts for the second term in the running time expression.

Our presentation is organized as follows. We first review the 2-color algorithm of [GT], and explain how $\bar{\delta}$ can be computed in this case. We then augment the 2-color algorithm of [GT] with lexicographic cost comparisons to help handle calls from our d-color algorithm. We finally present our recursive routine to find a d-color base.

The 2-color algorithm in [GT] is designed to find a minimum cost base constrained to have exactly s red elements, for some s. The algorithm calls a recursive routine to identify what is called a *restricted* swap sequence, which transforms a constrained minimum cost base of green elements to a constrained minimum cost base of red elements. The restricted swap sequence contains swaps in order of nondecreasing cost of the red element in each swap. The algorithm then sorts the swaps in order of nondecreasing cost of the swaps to yield an optimal swap sequence. The desired base is then formed by taking the first portion of the swap sequence and applying it to the green constrained minimum cost base. Since the cost of a minimum cost base with i red elements is a convex function of i, the vector $\bar{\delta}$ can be readily determined by comparing the cost of swaps adjacent to the desired base.

We augment the algorithm to enforce a lexicographic tie-breaking scheme. In addition to its color, let each element have a unique index. Assign a *tag* to each element consisting of the pair $(j, index)$, where j is the original color of the element. Ties in element costs are broken lexicographically using element tags. Ties in the costs of swaps are broken lexicographically as follows. Consider two swaps (e, f) and (e', f') of equal cost. Swap (e, f) will be lexicographically less than (e', f') if and only if either f or e' has the lexicographically smallest tag from among e, f, e', and f'. We can incorporate this lexicographic tie-breaking scheme into the 2-color algorithm of [GT] at constant cost for any comparison of two elements or two swaps.

We now describe our recursive routine to find a d-color base. The routine uses a divide-and-conquer approach, recursing first on fewer colors, and then again on fewer elements. The basis cases occur when either $d = 2$ or $n \leq d(2d-3)$. If $d = 2$ we use the augmented 2-color algorithm. We will discuss the other basis case later. If $d > 2$ and $n > d(2d-3)$, we do the following. Order the colors so that $q_j \leq q_{j+1}$, for $j = 1, 2, \cdots, d-1$. Find the constrained minimum cost base $B_{\bar{i}}$ where $i_j = q_j + \lfloor (q_d + j - 1)/(d-1) \rfloor$ for $j = 1, 2, \cdots, d-1$, and $i_d = 0$. This is a problem in $d-1$ colors, and is solved recursively by our routine. Note that $\bar{i}$ is defined so that for each color $j \neq d$, $B_{\bar{i}}$ has at least $\lfloor n/(d(d-1)) \rfloor$ more elements of color j than $B_{\bar{q}}$. Along with determining $B_{\bar{i}}$, the recursive call will supply the corresponding values $\delta(j)$, for $j = 1, \cdots, d-2$ that make $B_{\bar{i}}$ of minimum cost among bases with no elements of color d.

Once $B_{\bar{i}}$ and $\overline{\delta}$ have been determined, temporarily add $\hat{\delta}(j)$ to the cost of each element of color j in $B_{\bar{i}}$, for $j = 1, \cdots, d-2$. Define $\overline{f}$ such that for any d-tuple $\overline{i'}$, $f_j(i_j')$ $= |i_j' - i_j|$, for $j = 1, \cdots, d$. For any choice of π, $B_{\bar{i}}$ will be the minimum cost base among those with no elements of color d, with respect to the adjusted version of the cost function $c_L(\cdot)$, defined earlier.

Relabel the elements of base $B_{\bar{i}}$ with the color *green*, and label with the color *red* the elements in the constrained minimum cost base of color d. Now use the 2-color algorithm of [GT], augmented to use tags lexicographically to break ties in the costs of elements and swaps, to find the constrained minimum cost base B' which has $\lfloor q_d/(d-1) \rfloor - 1$ red elements and the rest green. Even though colors are reordered to satisfy $q_j \le q_{j+1}$, a permutation π can be chosen that undoes this reordering, and hence makes the use of the tags enforce $c_L(\cdot)$. Thus any base generated by the augmented 2-color algorithm will be a constrained minimum cost base with respect to $c_L(\cdot)$, and thus also $c(\cdot)$, in the original d-color matroid.

If we switch the elements in B' back to their original colors, we get a base $B_{\bar{k}}$ in which $k_d = \lfloor q_d/(d-1) \rfloor - 1$ and $k_j \ge q_j + 1$ for $j = 1, 2, \cdots, d-1$. It is clear that the set of color vectors consisting of $\overline{q}$ and its immediate neighbors dominate $\overline{k}$ with respect to color d. By our dominance theorem, every element of color d in $B_{\bar{k}}$ is in $B_{\bar{q}}$, and also in every constrained minimum cost base that is an immediate neighbor of $B_{\bar{q}}$. Contract the matroid on these elements of color d, and decrease q_d accordingly. Since $q_d \ge \lceil n/d \rceil$, the number of elements is reduced by at least $\lfloor \lceil n/d \rceil / (d-1) \rfloor - 1$, which is at least 1 if $n > d(2d-3)$. For $d > 2$ and $n > d(2d-3)$, note that the new value of q_d will be greater than 0. Solve the resulting smaller d-color problem recursively, and union its solution elements with the elements of color d already identified to give the complete set of solution elements. Take as the $\delta(j)$ values the values returned by this recursive call. This completes the procedure.

We justify the contraction and union steps in the previous paragraph as follows. Let D be the set of elements contracted, and M/D the contracted matroid. Note that $D \subset B_{\bar{q}}$, and $B_{\bar{q}}-D$ is a base in M/D. Let B be a base in M/D with the same index vector as $B_{\bar{q}}-D$ but not equal to $B_{\bar{q}}-D$. Now $c(B) > c(B_{\bar{q}}-D)$, since otherwise $B \cup D$ would be a base of M with index vector $\bar{q}$ but of smaller cost than $B_{\bar{q}}$, a contradiction to the definition of $B_{\bar{q}}$.

We now discuss the other basis case, when $n \leq d(2d-3)$. Here we use the weighted matroid intersection algorithm [L1, L2] to find $B_{\bar{q}}$ directly. We also need to determine the $\delta(j)$ values. This can be done by considering each of the elements not in $B_{\bar{q}}$. For each such element f, find the best swap in $B_{\bar{q}}$ for each color $j \neq color(f)$. We infer the values of $\delta(j)$ from the thresholds of these swaps as follows. Each best swap (e, f) yields a constraint $\delta(color(e)) - \delta(color(f)) \leq c(f) - c(e)$. Choosing the $\delta(j)$'s then reduces to the following shortest path problem. Consider a graph with d vertices labeled from 1 to d. For each constraint $\delta(j_1) - \delta(j_2) \leq c_{j_1 j_2}$ there is an edge from j_2 to j_1 of cost $c_{j_1 j_2}$. In the case of multiple edges, only the shortest edge is retained. Then choosing $\delta(j)$ to be the shortest distance from vertex d to vertex j, for all j, will give a consistent set of deltas. The shortest distances can be determined in $O(d^3)$ time using the Bellman-Ford algorithm in [L1]. This completes our presentation of the d-color algorithm.

We claim that the above algorithm solves any d-color problem. Since the number of elements contracted is at least 1 for $n > d(2d-3)$, the algorithm terminates. We next analyze the running time.

Theorem 4. Let M be a matroid of rank n with m elements of $d > 2$ colors. Let $T_0(m, n)$ be the time to solve the uncolored (monochromatic) problem, in M. Let $T(n, 2)$ the time to solve the 2-color problem in M with elements recolored to just 2 colors. If independence testing in M is polynomial, and the time to contract $O(dn)$ elements in M is $O(d\ T(n, 2))$, then the time to solve a d-color problem in M is $O(d\ T_0(m, n) + (d-1)!\ d!\ T(n, 2))$.

Proof. Let $T(n, d)$ be the time to solve a d-color problem in a matroid of rank n, given that the monochromatic bases are provided. The intersection algorithm in [L1, L2] uses $O(n^2 m(m+I(m)))$ time, where $I(m)$ is the time to test independence. By assumption, $I(m) = m^k$ for some k. Since $m = nd$, this takes $O(d^7(d^3+d^{3k}))$ time. Finding the swaps to identify $\delta(j)$ values involves examining $O(d^3)$ elements f, at $O(d^2)$ time per element f, or $O(d^5)$ time altogether. Thus we have $T(n, d) \le c_1\ d^{10}+d^{7+3k}$, for $n \le d(2d-3)$. For $n > d(2d-3)$, we have the recurrence

$$T(n, d) \le c_2\, nd + T(n, 2) + T(n, d-1) + C(n, d) + T(\lceil n(1-1/(d(d-1)))\rceil, d)$$

where the c_i's are constants, and $C(n, d)$ is the time required to contract a matroid consisting of the union of d monochromatic minimum cost bases of rank n, recovering a monochromatic minimum cost base of each color in the contracted matroid. Since

$T(n, 2) \geq n$. for $d \geq 2$ we have

$$T(n, d) \leq c_3((2\, d!\, (d-1)! - d)\, T(n, 2) + (d!\, (d-1)! - d)n)$$

for an appropriately chosen constant c_3. $\square$

We discuss the motivation for assuming the bound of $d\ T(n, 2)$ on the time to contract $O(dn)$ elements in a matroid. By assigning color $d+1$ to each element to be contracted and solving d 2-color problems involving color $d+1$ and each original color. we can determine the elements in each monochromatic base in the contracted matroid. The correctness of this reduction follows from the definition of matroid contraction. It is also necessary to determine the new attributes of each element (*e.g.*, endpoints of an edge in the case of a graphic matroid) in the contracted matroid. For all the matroids discussed in [GT], this can be done for each new base within time proportional to $T(n, 2)$.

Even though the running time involves factorials in terms of d, it is better than the running time for the weighted matroid intersection algorithm of [L1, L2] whenever d is $o((\log n)/(\log\log n))$.

We suggest a modification to the algorithm that may yield a faster algorithm in practice. The 2-color algorithm in [GT] generates in succinct form the sequence of constrained minimum cost bases between the base of all one color and all the other color. Instead of specifying the number of elements of color d that we want in B', we take the swap sequence generated, switch back to original colors and find the furthest base $B_{\bar{k}}$ represented in the swap sequence such that $k_j \geq q_j + 1$, for $j = 1, \cdots, d-1$. At least as many elements will be contracted as before.

Finally, as an illustration, we apply the above algorithm to graphic matroids. Here $T_0(m,n)$ is $O(m \log \beta(m,n))$ by the algorithm of [GGST], where $\beta(\cdot,\cdot)$ is a certain slowly growing function [FT]. $T(n,2)$ is $O(n \log n)$ by the algorithm of [GT]. Independence is equivalent to acyclicity, and thus independence can be tested in $O(m)$ time. Contracting $O(dn)$ elements can be implemented in $O(dn)$ time. We therefore have the time to find a constrained minimum cost spanning tree being $O(dm \log \beta(m,n) + (d-1)!d!n \log n)$.

5. Basic on-line update strategy

In this section we give a basic description of our data structures for on-line updating of a constrained minimum cost base in a d-color matroid. This work is an extension of the updating approach in [FS] which handled 2-color problems. Let $B_{\bar{i}}^{(h)}$ represent the minimum cost base for *colors* vector $\bar{i}$ after h element cost updates have been performed. We first discuss data structures that allow us to find quickly base $B_{\bar{i}}^{(h+1)}$ given $B_{\bar{i}}^{(h)}$ and all of its neighbors after h updates. This operation, which relies on Theorem 3, is crucial to our on-line update technique. However, to compute $B_{\bar{i}}^{(h+2)}$ by this method, we need to have $B_{\bar{i}}^{(h+1)}$ and its neighboring bases after $h+1$ updates, which in worst case means we must have $B_{\bar{i}}^{(h)}$, its neighbors after h updates, and also the neighbors' neighbors after h updates. We therefore discuss how to maintain larger groups of neighboring bases, and introduce the notion of an *arrangement* of bases, generalizing the sequences employed in the 2-color algorithm. Since updating large groups of bases directly would be quite inefficient, we then discuss maintaining arrangements in an implicit form, which allows for efficient updating. Finally, we illustrate the technique with the example of a graphic matroid. Although our presentation of the d-color update technique is sufficiently detailed to be self-contained, familiarity with the 2-color update technique of [FS] will greatly help in understanding the details.

We recall from [FS] the definition of an *update structure* for a base in a matroid with uncolored elements. An update structure for a base B is a data structure which supports the following operations:

maxcirc (f, B): finds the maximum cost element in the circuit $C(f, B)$.

mincocirc (e, B): finds the minimum cost element in the cocircuit $\overline{C}(e, B)$.

swap (e, f, B): converts the update structure for B into an update structure for $B - e + f$ (assuming that $f \in \overline{B}$ and $e \in C(f, B)$).

Let $U(m, n)$ represent the maximum of the execution times of these three operations for a particular matroid. Thus a minimum cost base in a matroid with uncolored elements can be updated in time at most $2U(m, n)$ when the cost of a single matroid element is modified. Let $S(m, n)$ be the space required by the update structure.

In the case of a matroid with elements of d colors, the update structure is generalized to allow the color of the appropriate element to be specified. Thus for $j = 1, 2, \cdots, d$, the operation *maxcirc* (j, f, B) finds the maximum cost element of color j in $C(f, B)$, and *mincocirc* (j, e, B) finds the minimum cost element of color j in $\overline{C}(e, B)$. The operation *swap* (e, f, B) is as before. The generalized update structure for d-colored matroids can be derived from the corresponding structure for uncolored matroids in a straightforward manner. For each field relating to costs in the uncolored update structure, maintain d fields in the new structure, with the j-th field accessed for operations on color j. The values in the fields should be such that the cost of an element not of color j should be treated as $-\infty$ in handling a *maxcirc* $(j,\cdot,\cdot)$, and ∞ in handling a *mincocirc* $(j,\cdot,\cdot)$.

Using Theorem 3, a generalized update structure can be used to find an updated base $B_{\bar{q}}^{(h+1)}$ from $B_{\bar{q}}^{(h)}$ and its neighbors after h updates. For instance, if a basic element e increased in cost, then $B_{\bar{q}}^{(h+1)}$ would be the least cost base in the set consisting of $B_{\bar{q}}^{(h)}$ and $B_{\bar{i}}^{(h)} - e + \mathit{mincocirc}\,(j, e, B_{\bar{i}}^{(h)})$, where either the color of e is not j, and $B_{\bar{i}}$ is a neighbor of $B_{\bar{q}}$ containing one fewer element of color j, or j is the color of e, and $B_{\bar{i}}$ is $B_{\bar{q}}$. If a cobasic element f decreased in cost, then $B_{\bar{q}}^{(h+1)}$ would be the least cost base in the set consisting of $B_{\bar{q}}^{(h)}$ and $B_{\bar{i}}^{(h)} - \mathit{maxcirc}\,(j, f, B_{\bar{i}}^{(h)}) + f$, where either the color of f is not j, and $B_{\bar{i}}$ is a neighbor of $B_{\bar{q}}$ containing one more element of color j, or j is the color of f, and $B_{\bar{i}}$ is $B_{\bar{q}}$. The update is concluded by performing the appropriate *swap*.

As stated at the beginning of the section, maintaining just $B_{\bar{q}}^{(h)}$ and its neighbors after h updates is not enough, since there is not sufficient information to compute efficiently all neighbors of $B_{\bar{q}}^{(h+1)}$ after $h+1$ updates. For $l > 0$, let $R_{\bar{i},l}$ be the set of bases $\{B_{\bar{i}'} \mid i_j' \le i_j + l - 1,\ j = 1, 2, \cdots, d\}$. We shall represent groups of bases in sets such as $R_{\bar{i},l}$, which we call *arrangements*. We say that arrangement $R_{\bar{i},l}$ is *centered on* $\bar{i}$ and has *radius* l. Our update procedure is periodic with period z. By this we mean that for the h-th element cost change the update procedure handles data in the same form (*e.g.*, radius of arrangement) as the data during the $(h+z)$-th element cost change, for any $h > 0$. Here, z is a parameter that will be specified later, when we discuss the running time. Our update procedure consists of three parts. For clarity, we will uncover the parts one by one.

Consider h to be an integer in the range from 0 to z. Suppose after the h-th update we keep an arrangement $A_0^{(h)} = R^{(h)}_{\bar{q},z-h}$. The superscript on R and on A_0 indicates how many element cost changes have been supplied, and will be omitted unless the context demands it. As long as $h < z$, there is sufficient information to generate $R^{(h+1)}_{\bar{q},z-h-1}$, no matter what type of element cost change occurs. Thus $z-1$ element cost changes can be successfully handled, but when the z-th update occurs, $B_{\bar{q}}$ is lost. This follows, since $A_0^{(z-1)}$ is an arrangement consisting of one base $B_{\bar{q}}^{(z-1)}$, so there is insufficient information remaining in order to compute $B_{\bar{q}}^{(z)}$. We say that A_0 *decays* during this sequence of z updates. Of course, for large z, explicitly maintaining and updating the arrangement A_0 requires considerable time per cost change. In due course, we will show how to circumvent this problem by introducing an implicit representation for A_0.

When A_0 has completely decayed, we need to replace it by an arrangement containing many bases. But this means that certain work must be done in advance. We therefore discuss the second part of our solution. We thus now consider unrestricted values of h. Whenever A_0 is initialized, *i.e.*, $h \bmod z = 0$, we initiate a computation to solve a number of d-color problems on the current matroid, in order to generate a new arrangement of bases, given the minimum cost base after h updates containing only elements of color j, for $j = 1, 2, \cdots, d$. Note that any constrained base after h updates contains only elements from the union of these monochromatic bases. Let $P(n, d)$ be the time required to determine for a given d-color problem an arrangement of bases in an appropriate form. Assume that copies of the d monochromatic bases are maintained from one update to the next. Since just one of these monochromatic bases changes, a cost of $U(m,n)$ is charged to the update. Each static d-color problem will be solved during the time in which A_0 decays, by performing $O(P(n,d)/z)$ work over each of z update steps.

However, when all static d-color problems are completed, after $h = kz$ updates, we cannot just reconstitute A_0 with the appropriate bases. This is because each such base will be *out-of-date* by z element cost changes, since the element costs used in solving the static problems were extracted after $(k-1)z$ updates, and z further element cost updates have been applied to the matroid in the meantime. Thus we introduce the third part of our update strategy. We use a second arrangement A_1, centered at $B_{\bar{q}}$ and initially with $l = 3z$, which is extracted from the out-of-date solution to the static d-color problems. Thus when A_1 is created after $h = kz$ updates have occurred, we have $A_1^{(h)} = R_{\bar{q},3z}^{(h-z)}$.

Since the bases in $A_1^{(kz)}$ will initially be out-of-date with respect to $A_0^{(kz)}$ by z element cost changes, we need to bring them up-to-date over the next z update steps of A_0, using the z element cost changes that have not yet been applied to A_1. These previous element cost changes can be saved in a queue as the static d-color problems are being solved. Thus, when $A_1^{(kz)}$ is created, the queue will contain element cost changes numbered $(k-1)z+1, (k-1)z+2, \ldots, kz$. Consider the h-th update step, that transforms $A_1^{(h-1)}$ to $A_1^{(h)}$. Let $h = kz+r$, where $0<r\leq z$. We first add the h-th element cost change to the rear of the queue. We then delete the two element cost changes

(namely, those numbered $h-z+r-1$ and $h-z+r$) from the front of the queue and apply them both to $A_1^{(h-1)}$, obtaining $A_1^{(h)}$. Thus $A_1^{(h)}$ will be the arrangement $R_{\bar{q},3z-2r}^{(h-z+r)}$. A_1 will then become up-to-date with respect to A_0, and also be of the correct radius, precisely when A_0 has completely decayed. We then replace A_0 by the current arrangement A_1.

We can view our three-part update technique as three concurrent processes going on at once. Times at which $h > 0$ and $h \bmod z = 0$ are regarded as *renewal points* for A_0. At a renewal point, A_0 has completely decayed, A_1 has caught up with A_0 and can replace it, the static d-color problems have completed from which a new A_1 can be constituted, and new static problems can be initiated.

We now discuss how to avoid the expense of repeatedly updating each base in the arrangements A_0 and A_1. We do this by maintaining an implicit representation of each arrangement. An *extremal base of color* j of arrangement $R_{\bar{q},l}$ is a base $B_{\bar{i}}$ where $i_j = q_j - (d-1)(l-1)$ and $i_{j'} = q_{j'} + l-1$ for $j' \neq j$. We denote this base as $B_{\bar{q},l,j}$. We also use the base $B_{\bar{q},l-1,j}$ and call this a *near-extremal base of color* j. For $g = 0, 1$ and $0 \leq r < z$, let $a = g(z-r)$, and $b = z-r+g(2z-r)$. For each arrangement $A_g^{(h)}$ with $h = kz+r$, $0 \leq r < z$ and $g = 0, 1$, except for when $g = 0$ and $r = z-1$, we maintain for each color j, $B_{\bar{q},b,j}^{(h-a)}$ and its j-positive neighbors, and $B_{\bar{q},b-1,j}^{(h-a)}$ and its j-negative neighbors. For $d = 3$, this amounts to four bases near (and including) each of three extremal bases, for a total of twelve bases. For $d > 3$, there will be $2d$ bases near (and including) each of d extremal bases, for a total of $2d^2$ bases. We call the set of these bases the *extreme bases*. For each extreme base we maintain its update structure. Using the algorithm from the previous section, each of the $2d^2$ bases can be found in $T(n,d)$ time, and thus $P(n,d)$ is $O(d^2T(n,d))$. One can actually find the set of bases faster, as we discuss in the proof of Theorem 5. All solutions to a 3-color problem with $n = 24$ are shown symbolically in Figure 3. An arrangement centered at $\overline{q} = (9, 8, 7)$ with radius $l = 4$ is shown in bold, with the extreme bases shown as the boldest.

We also maintain the set of all elements that are in some but not all bases of the arrangement, and call this set the *symmetric difference*. We bound the size of this set as follows. Consider some color j. By Theorem 2, all $q_j - (d-1)(l-1)$ elements of color j in $B_{\overline{q},l,j}$ will be in *every* base in the arrangement. Consider a base $B_{\overline{i}}$ which has $i_j = q_j + (d-1)^2(l-1)$ elements of color j and $i_{j'} = q_{j'} - (d-1)(l-1)$ elements of every color $j' \neq j$. Also by Theorem 2, *any* element of color j in *some* base in the arrangement will be in base $B_{\overline{i}}$. Subtracting, we infer that there will be at most $d(d-1)(l-1)$ elements of color j in the symmetric difference, or at most $d^2(d-1)(l-1)$ elements overall.

There are certain matroids (for instance, graphic matroids) for which update structures for bases in a contracted matroid can be maintained efficiently when elements are inserted and deleted. In such cases, we can save both space and time if we construct a contracted matroid for each arrangement. For each color j, we contract every element of color j that is guaranteed to be common to all bases in the arrangement. Thus for $j = 1, \ldots, d$ we contract all the j-colored elements in the extremal base of color j in the arrangement. The total number of elements contracted will be $\sum_{j=1}^{d}(q_j - (d-1)(l-1))$ $= n - d(d-1)(l-1)$. Since the contracted elements are independent in the original matroid, the resulting contracted matroid will have rank $d(d-1)(l-1)$. We also note that since the original matroid has a monochromatic base of each color, so will the contracted matroid. In what follows we will assume that, whenever appropriate, update structures are maintained for these smaller monochromatic bases in the contracted matroid.

To summarize, each update step h, where $h = kz + r$ and $0 \leq r < z$, involves the following operations. The monochromatic minimum cost base is updated for the color of the element whose cost has changed. The arrangement $A_0^{(h-1)}$ is transformed to $A_0^{(h)}$ by applying the h-th element cost change to it as follows. Depending on the type of element cost change, the new version of one of the bases near the extremal base of each color is computed. For each color j, either the extremal base $B^{(h)}_{\bar{q}, z-r, j}$ and its j-positive neighbors, or the near-extremal base $B^{(h)}_{\bar{q}, z-r-1, j}$ and its j-negative neighbors are used. If the cost of a basic element of color j' increases, then the new bases are generated using extremal bases and their j-positive neighbors. In this case the new bases will be $B^{(h)}_{\bar{q}, z-r, j'}$, and the (j', j)-neighbor of $B^{(h)}_{\bar{q}, z-r, j}$ for all $j \neq j'$. We have previously discussed how $B^{(h)}_{\bar{q}, z-r, j'}$ may be obtained from $B^{(h-1)}_{\bar{q}, z-r, j'}$ and its j'-positive neighbors. When $j \neq j'$, let B denote $B_{\bar{q}, z-r, j}$, and B' denote B's (j', j)-neighbor. Since the complete, positive, tight set consisting of B' and its j'-positive neighbors is identical to the complete, positive, tight set consisting of B and its j-positive neighbors, the sparse representation of the arrangement has sufficient information to generate the updated version of base B'.

If the cost of a nonbasic element of color j' decreases, then the near-extremal bases and their j-negative neighbors are used. In this case the new bases will be $B^{(h)}_{\bar{q}, z-r-1, j'}$, and the (j, j')-neighbor of $B^{(h)}_{\bar{q}, z-r-1, j}$ for all $j \neq j'$. The new symmetric difference is determined, as well as the contracted matroid (if maintained).

Let $W(n,d,z)$ be the time to perform the last two operations. A total of $2d^2+1$ static d-color problems of rank $n' = \Theta(d^2z)$ involving elements in the symmetric difference set are extracted in time $Q(n,d,z)$ (which is zero if the contracted matroid is maintained). Solving the $2d^2+1$ static d-color problems each in time $T(d^2z,d)$ then generates $B_{\bar{q}}^{(h)}$ and the extreme bases for the new arrangement $A_0^{(h)}$. The update structures for the extreme bases in $A_0^{(h-1)}$ are modified via swaps to yield update structures for these new bases, resp. We then have the implicit representation for $A_0^{(h)}$ after the update step.

Finally, $A_1^{(h-1)}$ is transformed to $A_1^{(h)}$. The h-th element cost change is added to the rear of the queue of element cost changes that we maintain for A_1. Two element cost changes from the front of the queue are then deleted and each is applied to $A_1^{(h-1)}$ in the same manner as the cost changes were applied to A_0, obtaining $A_1^{(h)}$.

Theorem 5. Let M be a matroid of rank n with m elements of d colors. Consider constrained minimum cost bases with respect to cost function $c_L(\cdot)$. The on-line update problem for such bases can be solved in $O(d^2U(m,n) + Q(n,d,z) + d^2T(d^2z,d) + W(n,d,z) + d\ T(n,d)/z + d^2T(d^2z,d)/z)$ time and $O(S(m,n))$ space.

Proof. For each of the $O(d^2)$ extreme bases of each arrangement, an update operation will be performed. Then d new extreme bases in each arrangement are selected from these $O(d^2)$ updated bases. An updated arrangement $A_g^{(h)}$ is generated by solving $O(d^2)$ static d-color problems. This can be done by first finding the extreme bases for each color on a contracted matroid of rank $n' = O(d^2z)$. Thus solving the static d-color problems will take time $O(d^2T(d^2z, d))$. Thus each update step in A_0 or A_1 will take

$O(d^2 U(m,n) + Q(n,d,z) + d^2 T(d^2 z,d) + W(n,d,z))$ time.

In addition, $O(d^2)$ static d-color problems of rank n must be solved over z updates. For each color j, compute the extremal bases of color j. Then contract the matroid to one of rank $n' = O(d^2 z)$. The remaining extreme bases can be found in the contracted matroid. The time spent per update step on solving these static d-color problems is $O((d\ T(n,d)+d^2 T(d^2 z,d))/z)$. □

To illustrate the above technique, we describe the construction of update structures for graphic matroids and analyze their efficiency. The update structure for a minimum spanning tree uses dynamic tree data structures [ST] and two-dimensional topology trees [F]. The former allows us to perform the operations *maxcirc* and *swap* in time $O(\log n)$. The latter allows us to perform the operations *mincocirc* and *swap* in time $O(\sqrt{m})$. Thus for this update structure $U(m,n) = O(\sqrt{m})$. The space used by the structures is $O(m)$.

A contracted matroid is maintained in the form of a contracted graph. A topology tree [F] is used to maintain a heap of the edges incident on each vertex of the contracted graph. Each such vertex corresponds to a tree-structured connected component of contracted edges from the current constrained minimum spanning tree. Since topology trees of size $d^2 z$ support *insert*, *delete*, *split* and *merge* operations in $O(\log(d^2 z))$ time, updating the contracted graph can be implemented efficiently. The time to solve a static d-color problem is $T(n,d) = O(d!\ (d-1)!\ n \log n)$. We therefore have the following theorem.

Theorem 6. Let G be a graph with n vertices, and with m edges of d colors. Consider constrained minimum spanning trees with respect to cost function $c_L(\cdot)$. The on-line update problem for such spanning trees can be solved in $O(d^2\sqrt{m} + d^{3/2}(d!)^2\sqrt{n}\ \log n)$ time and $O(m)$ space.

Proof. We have $U(m,n) = O(\sqrt{m})$, $T(n,d) = O(d!\,(d-1)!\ n\ \log n)$ and Q is equal to zero. $W(n,d,z)$ will be $O(d^2\log(d^2z))$. Each update step in the arrangements will take $O(d^2\sqrt{m} + d^3(d!)^2\ z\ \log(d^2z))$ time. We must also replenish the second arrangement by solving a number of static problems of rank n, which will cost $O(((d!)^2n\ \log n + d^3(d!)^2z\log(d^2z))/z)$ time per update. We choose $z = \Theta(n^{1/2}/d^{3/2})$. The space bound is evident from the data structures employed. □

6. A recursive representation of arrangements

We can achieve better update times by using a more complex representation of arrangements. Consider the example in the last section involving graphic matroids. We can use a two-level approach for representing A_0 and A_1. Consider update step h, where $h = kz+r$ and $0 \le r < z$. Recall that $A_0^{(h)} = R^{(h)}_{\bar{q},z-h}$, where $R_{\bar{i},l}$ is the set of bases $\{B_{\bar{i}'} \mid i_j' \le i_j+l-1,\ j = 1, 2, \cdots, d\}$. Arrangement A_0 was represented implicitly by the extreme bases, their associated data structures, the symmetric difference, and the contracted matroid (if maintained). On an update step in A_0, d new bases were determined, the symmetric difference and the contracted matroid were updated using them, and $2d^2+1$ static problems of rank $n' = O(d^2z)$ were solved to find the new extreme bases.

In our modified method, a base at each extreme is computed as before. However, instead of solving a number of static problems with respect to A_0 on each update step in

A_0, we do the following. We maintain smaller arrangements A_{0j}, $j = 2, 3, \cdots, d-1$, centered near the extreme bases of A_0, and two smaller arrangements A_{00} and A_{01} centered at $B_{\bar{q}}$. We call these smaller arrangements *subarrangements*. Only when the subarrangements A_{0j}, $j = 2, 3, \cdots, d+1$, decay to single bases are a number of static problems solved with respect to A_0. A_{00} and A_{01} are maintained to be able to access $B_{\bar{q}}$ meanwhile.

Let l_{0j} be the radius of subarrangement A_{0j}, $j = 0, 1, \cdots, d+1$. For $j = 2, \cdots, d+1$, A_{0j} will be centered on $\overline{q_j}$, where $q_{jk} = q_k-(d-1)(l_0-l_{0j})$ for $k = j$ and $q_{jk} = q_k+l_0-l_{0j}$ for $k \neq j$. Let y be a parameter to be specified subsequently. At a renewal point for A_{0j}, $l_{0j} = y$ if $j = 0$, $l_{0j} = 3y$ if $j = 1$, and $l_{0j} = 2y$ if $j = 2, 3, \cdots, d+1$. Each subarrangement is represented implicitly by its $2d^2$ extreme bases, their associated data structures, the symmetric difference, and the contracted matroid (if any). If the contracted matroid is maintained, the extreme bases are of rank $n' = \Theta(d^2 z)$; otherwise the bases are of rank n. After the A_{0j}, $j = 2, \cdots, d+1$, have decayed to radius 1, $2d^3$ static problems with $n' = d(d-1)(l-l_{0j})$ will be initiated to determine the extreme bases for the new A_{0j}, $j = 2, \cdots, d+1$.

At a renewal point for A_0, A_{00} will be up-to-date with respect to A_0. A_{0j}, $j = 1, 2, \cdots, d+1$, will be out-of-date with respect to A_{00} (and therefore A_0) by y element cost changes. Times at which $h \bmod z > 0$ and $h \bmod y = 0$ are regarded as renewal points for for A_{0j}, $j = 0, 1, \cdots, d+1$. At a renewal point for A_{00}, A_{00} has completely decayed. A_{01} has caught up with A_{00} and can replace it, arrangements A_{0j}, $j = 2, \cdots, d-1$, have caught up with A_{00} but have decayed to single bases, the $(d-1)2d^2$

static problems have completed, which yield the extreme bases for the new arrangements A_{0j}, $j = 1, \cdots, d+1$, and a new set of static problems can be initiated using the single bases from the previous A_{0j}, $j = 2, \cdots, d+1$. As before, two update steps in an out-of-date arrangement will be performed for every update step in A_0. We will assume that $z \bmod y = 0$, so that A_{0j}, $j = 1, 2, \cdots, d+1$, will catch up with A_0 precisely when A_0 reaches its next renewal point. Arrangement A_1 is represented in a similar fashion. Subarrangements A_{1j}, $j = 1, \cdots, d+1$, will initially be out-of-date with respect to A_1 by y element cost changes. Since A_1 is itself out-of-date with respect to A_0, four update steps will be performed in each of A_{1j}, $j = 1, \cdots, d+1$, for every update step in A_0.

We discuss how to perform an update in A_0. The update for A_1 is similar. For each of the extreme bases of A_0, an update operation is performed. Then d new extreme bases are selected from those $O(d^2)$ updated bases. The set of all elements that are in some but not all of these d bases are computed, as well as the contracted matroid. In addition, for all extreme bases of A_{0j} that are not extreme bases of A_0, an update operation is performed. For each group of d bases in this set, a new extreme base is computed. The sets of all elements that are in some but not all such bases of A_{0j}, for each $j = 0, 1, \cdots, d+1$, are computed, as well as the corresponding contracted matroids. A total of $2d^2+1$ static d-color problems of rank $n' = \Theta(d^2y)$ are solved for each of the $d+1$ subarrangements of A_0. From the extreme bases of the A_{0j} that correspond to extreme bases of A_0, swaps that transform the old extreme bases of A_0 into the the new extreme bases can be inferred. The new symmetric differences and contracted matroids for A_0 and its subarrangements can then be determined.

In addition, the following static problems must be solved over a sequence of updates. To generate the extreme bases for A_1, $O(d^2)$ static d-color problems of rank n must be solved over z updates. To generate the extreme bases for A_{gj}, $g = 0, 1$ and $j = 1, 2, \cdots, d+1$, $O(d^3)$ static d-color problems of rank $n' = \Theta(d^2 z)$ must be solved over y updates.

Theorem 7. Let G be a graph with n vertices, and with m edges of d colors. Consider constrained minimum spanning trees with respect to cost function $c_L(\cdot)$. The on-line update problem for such spanning trees can be solved in $O(d^2\sqrt{m} + d^{8/3}(d!)^2 n^{1/3}\log n)$ time and $O(m)$ space.

Proof. For each of the $O(d^2)$ extreme bases of arrangements A_0 and A_1, an update operation will be performed. Then d new extreme bases in each arrangement are selected from these $O(d^2)$ updated bases. The time required is $O(d^2U(m,n))$. For each of the $O(d^3)$ extreme bases of subarrangements A_{0j} and A_{1j}, an update operation will be performed. Then d new extreme bases in each subarrangement are selected from its $O(d^2)$ updated bases. The total time required is $O(d^3U(d^2z,n))$. An updated arrangement A_{kj} is generated by solving $O(d^2)$ static d-color problems. This can be done by finding the extreme bases for each color on a contracted matroid of rank $n' = O(d^2y)$. Thus solving the static d-color problems will take time $O(d^3T(d^2y, d))$. Thus each update step in the arrangements and subarrangements will take $O(d^2U(m,n) + Q(n,d,z) + d^3U(d^2z,n) + d^3\,T(d^2y,d) + W(n,d,z))$ time.

In addition, $O(d^2)$ static d-color problems must be solved over z updates. As in the proof of Theorem 5, this will take $O((d\ T(n,d)+d^2T(d^2z,d))/z)$ time.

Also, $O(d^3)$ static d-color problems must be solved over y updates. The time spent per update step on solving these static d-color problems will be $O((d^3T(d^2z,d))/y)$.

The time spent handling each element cost change is $O(d^2\sqrt{m} + (d!)^2((n \log n)/z + d^4(z \log z)/y + d^4 y \log y))$. Choosing $z = \Theta(n^{2/3}/d^{8/3})$ and $y = \Theta(n^{1/3}/d^{4/3})$ yields the theorem. $\square$

For fixed d, the time for the above approach is limited by the $O(\sqrt{m})$ time to update a minimum spanning base in an uncolored graph. If the graph is planar however, then the update time in an uncolored graph has been shown to be $O(\log n)$ in [GS], and hence not a limiting factor. We thus extend recursively the implicit representation of arrangements. The representations will be of two types, centered and uncentered. Let $a(d)$ be a value depending on d, which we shall specify subsequently. An arrangement, centered or uncentered, of radius at most $a(d)$, is the set of extreme bases, their associated data structures, the symmetric difference, and the contracted matroid. Let $f(\cdot)$ be a function to be defined subsequently. For an arrangement A_λ of radius l_λ initially equal to $z > a(d)$, a *centered representation* consists of the above items, plus:

1. a centered representation of a subarrangement $A_{\lambda 0}$, which is centered on the same position as A_λ, with radius $l_{\lambda 0}$ initially equal to $f(z)$, and which is up-to-date with respect to A_λ.

2. a centered representation of a subarrangement $A_{\lambda 1}$, which is centered on the same position as A_λ, with radius $l_{\lambda 1}$ initially equal to $3f(z)$, and which is out-of-date with respect to A_λ by $l_{\lambda 0}$ element cost changes.

3. uncentered representations of subarrangements $A_{\lambda j}$, $j = 2, \cdots, d+1$, which are positioned at the extremes of A_λ, with radius $l_{\lambda j}$ initially equal to $2f(z)$, and which are out-of-date with respect to A_λ by $l_{\lambda 0}$ element cost changes.

4. $2d^2$ static problems which have just been initiated. Of these, $2d$ will be of rank $n' = \Theta(d^2 z)$, and the remainder of rank $O(d^2 f(z))$.

An *uncentered representation* consists of all items in a centered representation except items 1 and 2.

Let $f^{(0)}(x) = x$ and $f^{(i)}(x) = f(f^{(i-1)}(x))$, for $i > 0$. Then we choose the function $f(\cdot)$ such that $f^{(i+1)}(n) \bmod f^{(i)}(n) = 0$ for $i > 0$. This can be done easily by forcing $f(\cdot)$ to be a power of 2. This choice of $f(\cdot)$ ensures that each $(i+1)$-st level arrangement will have caught up with the appropriate i-th level arrangement at an i-th level renewal point.

Let $T_C(z)$ and $T_U(z)$ be the update times for centered and uncentered arrangements of radius z, respectively. The update times are described by the recurrences:

$$
\begin{aligned}
T_U(z) &= cd^3 d!\,(d-1)!\,(z \log z)/f(z) + 2dT_U(2f(z)) \\
T_C(z) &= cd^3 d!\,(d-1)!\,(z \log z)/f(z) + 2dT_U(2f(z)) \\
&\qquad + 2T_C(3f(z)) + T_C(f(z))
\end{aligned}
$$

where c is a constant. The first term in each recurrence represents the time spent per update step on solving the static problems of rank $\Theta(d^2 z)$ and updating the data structures. The remaining terms represent the time for recursively updating subarrangements of radius $\Theta(f(z))$, and reflect the fact that two update steps are required for out-of-date subarrangements for each update step in the primary arrangement.

Theorem 8. Let G be a planar graph with n vertices, and edges of d colors. Consider constrained minimum spanning trees with respect to cost function $c_L(\cdot)$. The on-line update problem for such spanning trees can be solved in $O(d^2(d!)^2(\log d)^{-1/2}$ $2^{\sqrt{2\log(2d)\log n}}(\log n)^{3/2})$ time and $O(n)$ space.

Proof. We have $U(m,n) = O(\log n)$, $P(n) = O(n \log n)$ and $Q = 0$. If we choose $f(x) = \Theta(x/2^{\sqrt{2\log(2d)\log x}})$ and observe that $\sqrt{\log f(x)} = \sqrt{\log x - \sqrt{2\log(2d)\log x}} < \sqrt{\log x} - \sqrt{(\log(2d))/2}$, then both $T_U(n)$ and $T_C(n)$ are $O(d^2(d!)^2(\log d)^{-1/2}\ 2^{\sqrt{2\log(2d)\log n}}(\log n)^{3/2})$, provided $a(d)$ is small enough, so that the basis of the recurrences satisfies these bounds.

For the space, the recursive representation has $(d+2)^i$ subsequences each using structures of size $\Theta(f^{(i)}(n))$ at level i. With $d+2 \le 2d \le 2^{\sqrt{2\log(2d)\log n}}$, the sizes of these structures sum to $O(n)$ over all levels. Solving for n in the above inequality suggests the choice of $a(d) = \sqrt{2d}$; since arrangements of radius at most $a(d)$ are represented explicitly, the space bound is as claimed. If the general matroid intersection algorithm is used for updating arrangements of radius at most $a(d)$ in the centered and uncentered representations, then the basis in the recurrences is polynomial in d. Thus the basis satisfies the claimed bounds on $T_U(n)$ and $T_C(n)$. ☐

7. Applications

The techniques of section 4 can be used to solve the minimum spanning tree problem when d vertices have degree constraints. Assume that the vertices with degree constraints are indexed $v_1, v_2, \cdots, v_d$. Label each edge incident on two constrained vertices with color 0. Label each edge incident on exactly one constrained vertex v_i with color i. Label each edge incident on two unconstrained vertices with color $d+1$.

Since there are d constrained vertices, there are at most $d(d-1)/2$ edges of color 0. In turn we consider every subset of edges of color 0 that is a forest, such that the degree of each v_i in the forest does not exceed its degree requirement r_i. We generate a candidate solution for each such forest. The idea is to include all the forest edges in the solution and then choose remaining edges so as to satisfy the degree constraints in a minimum cost fashion. The minimum cost solution over all such forests is then the minimum spanning tree satisfying the degree constraints.

For each forest, we generate a reduced graph as follows. Make a copy of the graph, and initialize r_i' to be r_i for $i=1, 2, \cdots, d$. Delete from the graph all edges of color 0 which are not in the forest. For each edge (v_i, v_j) in the forest, decrease by 1 the degree requirements r_i' and r_j'. Then contract the remaining edges of color 0 in the graph. To get the candidate solution corresponding to this forest solve a $(d+1)$-color static problem on the reduced graph, where r_i' edges of color i are desired, for $i=1, 2, \cdots, d$, and the remaining edges are of color $d+1$.

Theorem 9. The time to solve a minimum spanning tree problem with degree constraints on d of the vertices is $O(T_0(m, n) + d!\,(d+1)!\,(ed+2)^{d-1}T(n, 2))$.

Proof. The time to solve a minimum spanning tree problem on edges of color $d+1$ is $T_0(m, n)$. The number of forests is less than

$$\binom{d(d-1)/2}{d-1} < (d(d-1)/2)^{d-1}/((d-1)/e)^{d-1} = (ed/2)^{d-1}.$$

For each forest, a $(d+1)$-color problem is then solved. □

References

[B] O. Boruvka, *O jistem problemu minimalnim*, Praca Moravske Prirodovedecke Spolecnosti 3 (1926) 37-58.

[F] G. N. Frederickson, *Data structures for on-line updating of minimum spanning trees, with applications*, SIAM J. Computing 14 (1985) 781-798.

[FS] G. N. Frederickson and M. A. Srinivas, *On-line updating of solutions to a class of matroid intersection problems*, to appear in Information and Computation 74 (1987).

[FT] M. Fredman and R. E. Tarjan, *Fibonacci heaps and their uses in improved network optimization algorithms*, Proc. 25th Symp. on Foundations of Computer Science, Singer Island, Florida (1984) 338-346.

[GGST] H. N. Gabow, Z. Galil, T. H. Spencer and R. E. Tarjan, *Efficient algorithms for finding minimum spanning trees in directed and undirected graphs*, Combinatorica 6 (1986) 109-122.

[GS] H. N. Gabow and M. Stallmann, *Efficient algorithms for graphic matroid intersection and parity*, Proc. of International Conf. on Automata, Languages, and Programming, Napflion, Greece, Lecture Notes in Computer Science, Vol. 194, Springer-Verlag (1985) 210-220.

[GT] H. N. Gabow and R. E. Tarjan, *Efficient algorithms for a family of matroid intersection problems*, J. Algorithms 5 (1984) 80-131.

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

[G] D. Gusfield, *Matroid optimization with the interleaving of two ordered sets*, Disc. Appl. Math. 8 (1984) 41-50.

[L1] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, New York, 1976.

[L2] E. L. Lawler, *Matroid intersection algorithms*, Math. Programming 9 (1975) 31-56.

[ST] D. D. Sleator and R. E. Tarjan, *A data structure for dynamic trees*, J. Comp. Sys. Sci., 26 (1983), 362-391.

[W] D. J. A. Welsh, *Matroid Theory*, Academic Press, New York, 1976.

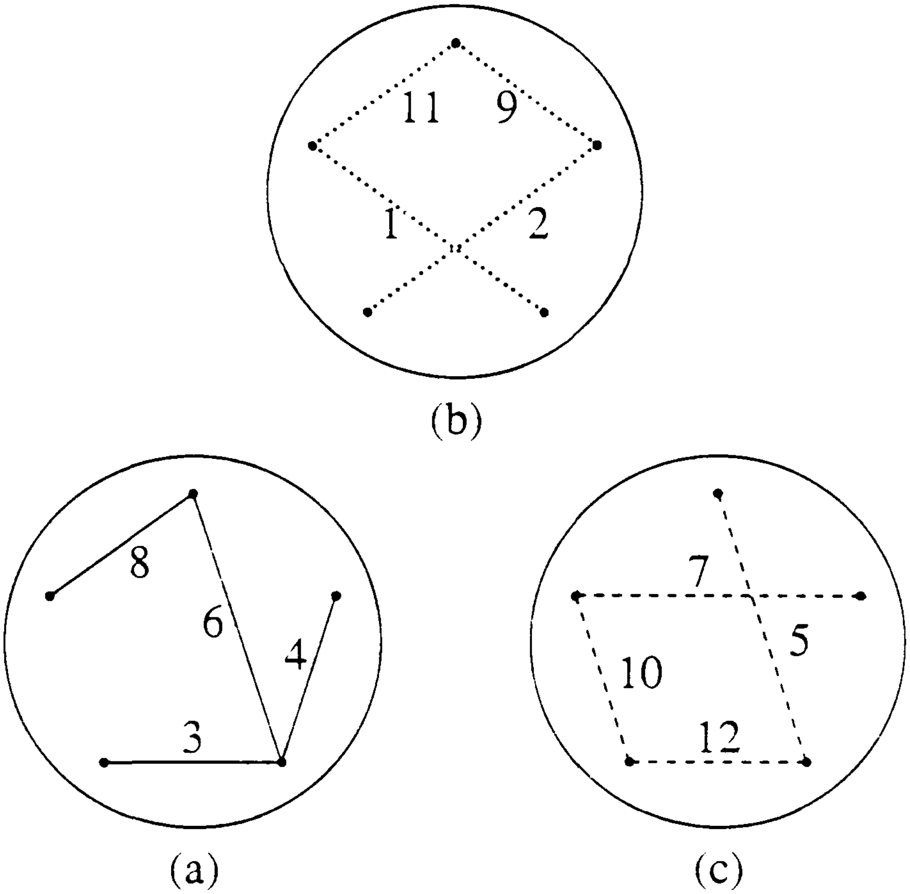


Figure 1. Subgraphs of a weighted graph with edges of three colors:
a. subgraph of red edges (solid lines)
b. subgraph of blue edges (dotted lines)
c. subgraph of green edges (dashed lines)

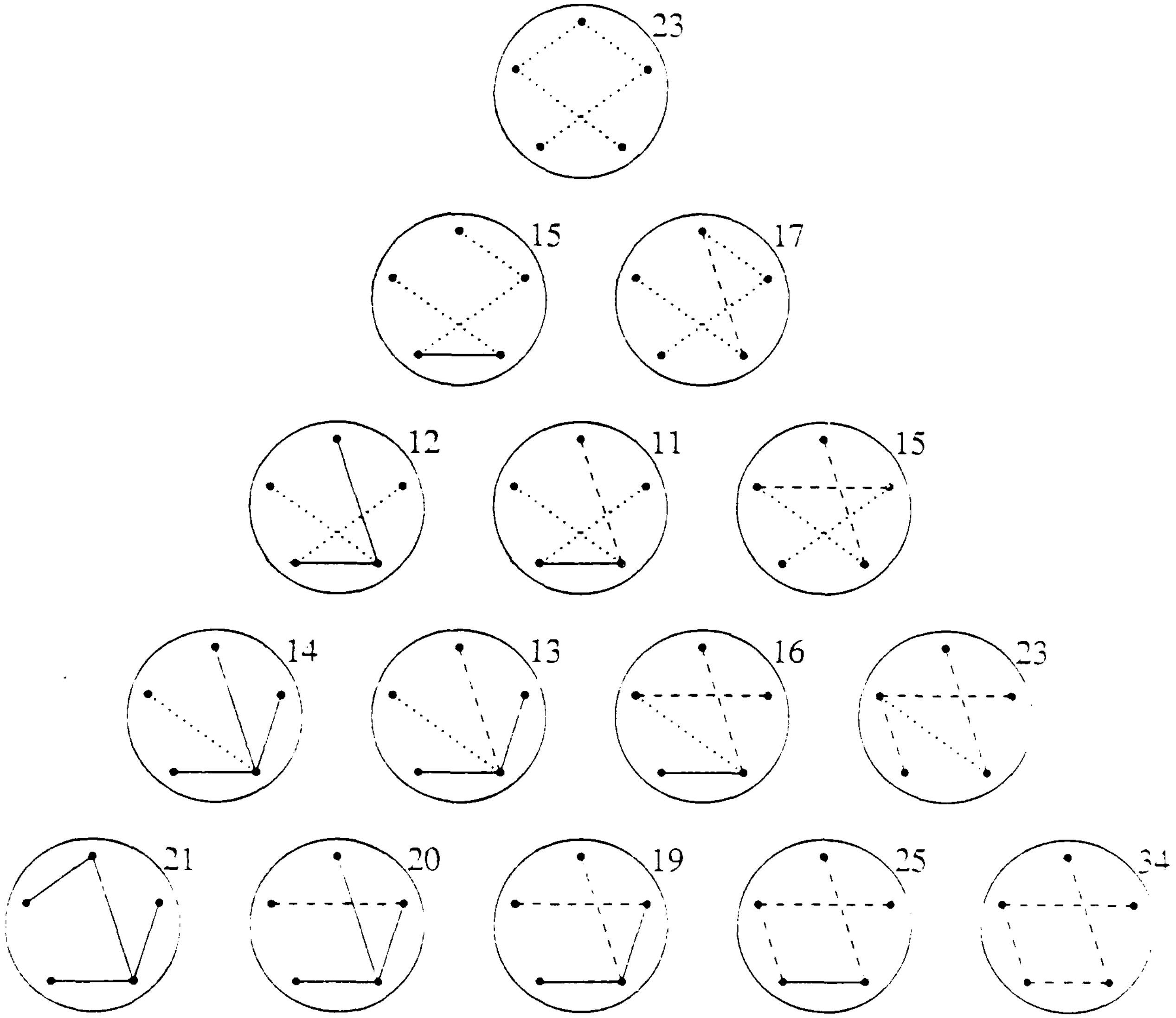


Figure 2. Solutions to all minimum spanning tree problems from Figure 1:
The tree with i red edges, j green edges, and $4-i-j$ blue edges
is the $(j+1)$-st tree in the $(i-j+1)$-st row from the top.

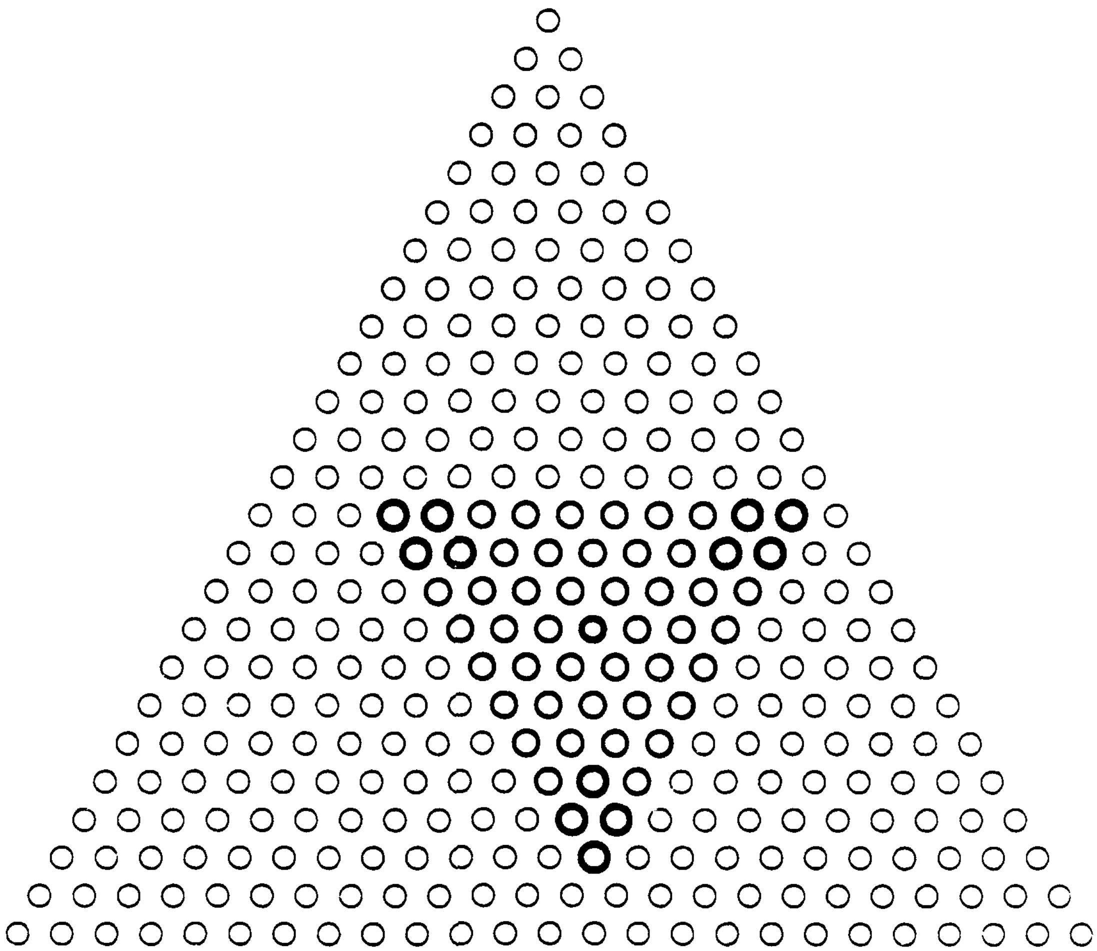


Figure 3. All solutions to a 3-color problem of 25 vertices.
Arrangement centered at $\overline{q} = (9, 8, 7)$ with radius $l = 4$ is in bold,
and the extreme bases are the boldest of the bases.

END
DATE
FILMED
5-88
DTIC