Multi-source Spanning Trees: Algorithms for Minimizing Source Eccentricities

Brendan McMahan* Andrzej Proskurowski†

November 15, 1999

Abstract

We present two efficient algorithms constructing a spanning tree with minimum eccentricity of a source, for a given graph with weighted edges and a set of source vertices. The first algorithm is both simpler to implement and faster of the two. The second approach involves enumerating single-source shortest-path spanning trees for all points on a graph, a technique that may be useful in solving other problems.

1 Introduction

Our problem is one of a family of problems where the goal is to find a "good" spanning tree of a graph for the purpose of communication from some subset of the vertices. Let $G = \langle V, E\rangle$ be a graph with a length function ℓ on edges and k source vertices $S = \{s_1, \ldots, s_k\} \subseteq V$. One defines "good" by providing a cost function c mapping spanning trees of G to $\mathcal{N}$. The most obvious metric is perhaps

$$c(T) = \sum_{s\in S}\sum_{v\in V} d_T(s,v),$$

where $d_T(s,v)$ is the length of the path from s to v in the spanning tree T of G. Finding a tree that minimizes this cost function is called the k-SPST

*Research supported in part by an REU supplement grant NCR 97148D; Whitman College, Walla Walla, WA 99362

†Research supported in part by an NSF grant NCR 9714680; University of Oregon, Eugene, OR 97403 USA

problem in [1] and shown to be NP-hard . That paper also introduces the k-MEST problem, where the cost function is defined as

$$c(T) = \max_{s \in S}\{ \max_{v \in V}\{d_T(s, v)\} \}.$$

While [1] presents a solution alorithm for instances of k-MEST with polynomially bounded integral length function ℓ, the algorithm is potentially exponential for general weights. We will give efficient solution for the general problem. Our presentation is as follows. In Section 2, we make several observations about solutions to this problem that lead to a fast approximation algorithm and then to a polynomial time algorithm for an exact solution. In Section 3, we discuss a technique for enumerating certain single-source shortest-path spanning trees (SPSTs) of a graph. We conclude with Section 4, in which we outline some future work.

We use the term "point" to describe a location on a geometric model of a graph. A point can be either a vertex or a location on an edge. Let α be a point on an edge (p, q). Then, $d_p(\alpha)$ is the distance along the edge from p to α, $d_q(\alpha)$ is the distance along the edge from q to α, and $d_p(\alpha)+d_q(\alpha) = \ell(p, q)$ (Thus, when defining a point α on an edge (p, q) it is sufficient to give either $d_p(\alpha)$ or $d_q(\alpha)$).

It is natural to think of an interval of points on an edge. We define such sets of points by analogy to intervals on the real line. Given points α_1 and α_2 on an edge (p, q) with $d_p(\alpha_1) < d_p(\alpha_2)$, a point β on (p, q) is in the open interval (α_1, α_2) if $d_p(\alpha_1) < d_p(\beta) < d_p(\alpha_2)$. Half-open and closed intervals are defined in an analogous manner. A vertex is trivially a point on all incident edges and so we have a dual interpretation for (p, q) as both an element of the set E and as the set of points on the edge, not including the endpoints.

Given a point α on an edge (p, q), we can modify the graph by replacing (p, q) with a path of two edges, say (p, v) and (v, q), where v is a new vertex. Call this new graph G'. We then define a length function ℓ' on G' that is equal to ℓ except that $\ell'(p, v) = d_p(\alpha)$ and $\ell'(v, q) = d_q(\alpha)$. An optimal solution to k-MEST on G' that contains both (p, v) and (v, q) will correspond to an optimal solution on G. This observation allows us to treat points as vertices when convenient.

The distance between a point α on (p, q) and a vertex v is clearly

$$\min\{d_G(p, v) + d_p(\alpha),\ d_G(q, v) + d_q(\alpha)\}.$$

Using this definition, the concept of a shortest path spanning tree (SPST) rooted at a vertex can be extended to an SPST rooted at point $\alpha \in (p, q)$. We will interpret these trees as spanning trees of G, requiring that they always include both (p, α) and (α, q) (true when there are no "long" edges, for which $d_G(p, q) < \ell(p, q)$).

Certain points on an edge (p, q) play an important role in the remainder of this paper, so we introduce them now. For each vertex $v \in V$, define the point γ_v on (p, q) such that for any point $\alpha \in (p, \gamma_v)$ the shortest path from α to v is through the vertex p, and for $\alpha \in (\gamma_v, q)$ the shortest path from α to v is through q. The location of these points on (p, q) is given by

$$d_p(\gamma_v) = \frac{1}{2}\left(d_G(q, v) - d_G(p, v) + \ell(p, q)\right). \qquad (1)$$

The points γ_p and γ_q on (p, q) as defined by Equation (1) are such that γ_p is located at the vertex q and γ_q is located at the vertex p except for the degenerate case where the edge (p, q) is "long" and there is a path between the two vertices shorter than $\ell(p, q)$. To simplify our arguments we will assume without loss of generality that there are no such "long" edges in G. The following theorem gives another result about points γ_p and γ_q . It is trivially true when we remove "long" edges, but the fact that it is true in general gives insight into the distribution of the points γ_v on (p, q).

Theorem 1 *On an edge (p, q), for any $v \in V \backslash \{p, q\}$, $d_p(\gamma_q) \le d_p(\gamma_v) \le d_p(\gamma_p)$.*

Proof: The triangle inequality for shortest paths gives $d_G(v, p) \le d_G(p, q) + d_G(q, v)$, or rearranging, $d_G(q, v) \ge d_G(p, v) - d_G(p, q)$. We can substitute this inequality into Equation 1 and conclude,

$$d_p(\gamma_v) \ge \frac{1}{2}(d_G(p, v) - d_G(p, q) - d_G(p, v) + \ell(p, q)) = d_p(\gamma_v).$$

A similar argument shows $d_p(\gamma_v) \le d_p(\gamma_x)$. $\square$

2 A Simple and Fast Algorithm

An important theorem about the structure of an optimal tree for the k-MEST problem is proved in [1]. We restate this theorem using the terminology just developed, and revisit the proof, as it leads to the development of a polynomial-time algorithm for k-MEST.

Theorem 2 *There exists a point χ on the graph G such that any SPST rooted at χ and interpreted as a spanning tree of G is a solution to the given k-MEST problem.*

We consider the following situation in this section. Let G, S, and ℓ be an instance of k-MEST as described in the introduction. Let T^* be a solution to the problem, that is T^* minimizes $\max_{s \in S, v \in V}\{d_T(s, v)\}$ over all possible spanning trees T. Let $\max_{s \in S, v \in V}\{d_{T^*}(s, v)\} = M^*$. Let q be the maximum intra-source distance in T^*, and let P be a path of length q in T^* between sources s_1 and s_2. Let χ be the midpoint of P, that is, $d_{T^*}(\chi, s_1) = d_{T^*}(\chi, s_2) = q/2$. As shown in [1], the shortest-path tree rooted at χ, T_χ is optimal. We restate part of this proof because it leads to a useful observation.

Proof of Theorem 2
First, we observe that $\forall s \in S, d_{T^*}(\chi, s) \leq q/2$, for otherwise there is an intra-source path of length greater than q. Also, $\forall v \in V,\ d_{T^*}(\chi, v) \leq M^* - q/2$, for otherwise some source (either s_1 or s_2) has eccentricity greater than M^*. The definition of T_χ as a shortest path tree implies $d_{T_\chi}(\chi, v) \leq d_{T^*}(\chi, v)$ for arbitrary $v \in V$. Using these facts, for arbitrary source $s \in S$ and vertex $v \in V$, we have

$$\begin{aligned} d_{T_\chi}(v, s) &\leq d_{T_\chi}(\chi, s) + d_{T_\chi}(\chi, v) \\ &\leq d_{T^*}(\chi, s) + d_{T^*}(\chi, v) \\ &\leq q/2 + (M^* - q/2) = M^* \end{aligned} \tag{2}$$

which shows that T_χ is also an optimal tree and so the theorem is proved. □

This proof immediately leads to an efficient approximation algorithm.

Theorem 3 *There is a vertex $\chi' \in V$ such that any SPST $T_{\chi'}$ rooted at χ' has maximum source eccentricity less than or equal to $2M^*$*

Proof: Let χ' be a vertex in G located at distance d from χ. We note that $d_{T_{\chi'}}(\chi', v) \leq d_{T_\chi}(\chi, v) + d$ for any vertex $v \in G$. Thus, for any source s and arbitrary vertex v, we have

$$\begin{aligned} d_{T_{\chi'}}(s,v) &\leq d_{T_{\chi'}}(s,\chi') + d_{T_{\chi'}}(\chi',v) \\ &\leq d_{T_\chi}(s,\chi) + d_{T_\chi}(\chi,v) + 2d \\ &\leq M^* + 2d \end{aligned}$$

where the last inequality is a consequence of Equation (2). This shows that if we can find a vertex sufficiently near χ, then an SPST rooted in that vertex will be a reasonable approximation to the optimal spanning tree T^*. We note that χ lies on an edge of T^*, and every edge of T^* has length no more than M^*. Thus, some vertex v in G is within distance $M^*/2$ of the point χ. Thus, if we construct an SPST for each $v \in V$ we are guaranteed to find a tree with maximum source eccentricity less than $2M^*$. □

The next theorem leads to a polynomial time algorithm for an exact solution. Let $\mathcal{G}$ be the set of all the points on the graph G. We define a function $R : \mathcal{G} \to \mathcal{R}$ as follows:

$$R(\alpha) = \max_{v\in V}\{d_G(\alpha,v)\} + \max_{s\in S}\{d_G(\alpha,s)\}.$$

Theorem 4 *If a point $\chi' \in \mathcal{G}$ minimizes $R(\alpha)$ over all $\alpha \in \mathcal{G}$, then $R(\chi') = M^*$.*

Proof: By Equation (2), for any $v \in V$ and $s \in S$, we have $d_{T_\chi}(\chi,s) + d_{T_\chi}(\chi,v) \leq M^*$. We note that T_χ is a shortest path tree so distances in T_χ from χ are also distances in G. Thus, we conclude

$$\max_{v\in V}\{d_G(\chi,v)\} + \max_{s\in S}\{d_G(\chi,s)\} \leq M^*.$$

Using this and the fact that T_χ is an optimal tree,

$$M^* = \max_{s\in S, v\in V}\{d_{T_\chi}(s,v)\} \leq \max_{v\in V}\{d_G(\chi,v)\} + \max_{s\in S}\{d_G(\chi,s)\} \leq M^*.$$

This immediately implies that

$$\max_{v\in V}\{d_G(\chi,v)\} + \max_{s\in S}\{d_G(\chi,s)\} = M^*, \tag{3}$$

or $R(\chi) = M^*$. For an arbitrary point α in $\mathcal{G}$ we must have

$$\begin{aligned} M^* &\leq \max_{s\in S, v\in V}\{d_{T_\alpha}(s,v)\} \\ &\leq \max_{v\in V}\{d_{T_\alpha}(\alpha,v)\} + \max_{s\in S}\{d_{T_\alpha}(\alpha,s)\}. \end{aligned} \tag{4}$$

By definition, distances from α in T_α are also distances in G, and so for arbitrary $\alpha \in \mathcal{G}, R(\alpha) \geq M^*$.

Suppose χ' is a point that minimizes $R(\alpha)$ over all points α in $\mathcal{G}$. We know there is a point such that $R(\alpha) = M^*$, namely the special point χ described above. Since χ' minimizes $R(\alpha)$, we must have $R(\chi') \leq M^*$. But we have shown that $R(\alpha) \geq M^*$ for all α. Thus, we conclude that $R(\chi') = M^*$. $\square$

Corollary 1 *A SPST rooted at χ' (a point that minimizes $R(\alpha)$ over all points $\alpha \in \mathcal{G}$) is an optimal tree for the k-MEST problem.*

Proof: This is an immediate consequence of Theorem 4 and Equation (4). $\square$

2.1 The Algorithm

We now describe an efficient algorithm to find a point that minimizes $R(\alpha)$. First, we compute all-pairs shortest paths (in $\mathcal{O}(|V|^3)$ time). Then, for each edge (p, q) we use this information to minimize $R(\alpha)$ on the closed interval $[p, q]$.

Consider the point α in the interval $[p, q]$ (where $(p, q) \in E$). It is clear that for any $v \in V$, $d_G(\alpha, v) = \min\{d_G(v, p) + d_p(\alpha), d_G(v, q) + d_q(\alpha)\}$. The point in $[p, q]$ at which the distance to v is maximized will then be the point where both of these quantities are equal. This is the point γ_v defined by Equation (1). We can calculate the location of these points for all $v \in V$ for the edge (p, q) in $\mathcal{O}(|V|)$ time. For each γ_v on $[p, q]$, let $d_v = d_G(p, v) + d_p(\gamma_v) = d_G(q, v) + d_q(\gamma_v)$, the distance from γ_v to v. We can now define the distance from an arbitrary point α in $[p, q]$ to v by the formula

$$d_G(\alpha, v) = d_v - |d_p(\gamma_v) - d_p(\alpha)|.$$

We have $|V|$ such equations for each edge, one for each vertex. For a point α in $[p, q]$, we now have

$$R(\alpha) = \max_{v \in V}\{d_v - |d_p(\gamma_v) - d_p(\alpha|)\} + \max_{s \in S}\{d_s - |d_p(\gamma_s) - d_p(\alpha)|\}. \tag{5}$$

Minimizing this function on the edge is effectively a problem in computational geometry, which can be solved in $\mathcal{O}(|V| \log |V|)$ time.

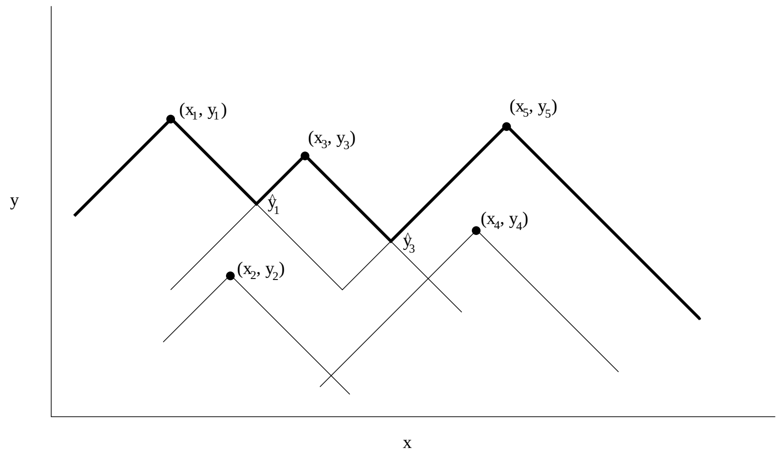

Figure 1: A sawtooth function

We first show how to solve this key subproblem. Let $P = \{p_1, \ldots, p_n\}$ be a set of points $p_i = (x_i, y_i)$ and all $x_i \in [0, \ell]$ are in non-decreasing order. Then, we wish to find a point $x \in [0, \ell]$ that minimizes

$$f(x) = \max_{1 \leq i \leq n} \{y_i - |x - x_i|\}. \tag{6}$$

The function $f(x)$ is a "sawtooth" function, where some of the points p_i define local maxima ("tips of the teeth") and any other point is "in the shadow" of a tip point. We call these points relevant and irrelevant, respectively. More precisely, a point (x_i, y_i) is *relevant* if $y_i - |x - x_i| = f(x)$ for some $x \in [0, \ell]$ and em irrelevant if $y_i - |x - x_i| < f(x)$ for all $x \in [0, \ell]$. An example of such a sawtooth function is shown in Figure 1. The thicker line shows the value of f. The points (x_1, y_1), (x_3, y_3), and (x_5, y_5) are relevant, while (x_2, y_2) and (x_4, y_4) are not. The three relevant points define two possible minimum values ("the valleys"), with y values $\hat{y}_1$ and $\hat{y}_3$.

To find minimum values of $f(x)$ we first remove irrelevant points. We do this in two stages. First we remove all points that are "in the shadow" of a point to their left. Let $p_{s_1}, p_{s_2}, \ldots, p_{s_n}$ be the points in non-decreasing order of $x_i + y_i$. This corresponds to sorting the points based on their projections onto the line $y = x$ via the line of slope -1. A point p_{s_i} is irrelevant if there is some $j > i$ such that $s_i \geq s_j$. We can find the points to remove in $\mathcal{O}(n)$ time by scanning from s_n to s_1. We keep track of the minimum index s_i seen so far, and discard the next point we scan if it has a greater index.

Removing the irrelevant points that are in the shadow of a point to their right is done in a similar manner, by letting $p_{t_1}, p_{t_2}, \ldots, p_{t_n}$ be a sequence of points in non-decreasing order of $y_i - x_i$, and removing a point p_{t_i} if there is a $j > i$ such that $t_i \geq t_j$.

Let the set of relevant points be $p_1^*, \ldots, p_r^*$. We intersect the line of slope -1 through p_i^* with the line of slope 1 through p_{i+1}^* for $1 \leq i \leq r-1$. These points of intersection are the points in the valleys of the sawtooth function, and hence are possible minimums. The heights of the valleys are given by

$$\hat{y}_i = \frac{1}{2}(x_i^* - x_{i+1}^* + y_{i+1}^* + y_i^*).$$

We denote the valley point immediately to the right of $p_i^* = (x_i^*, y_i^*)$ by $\hat{p}_i = (\hat{x}_i, \hat{y}_i)$. There are at most n of these valleys, so we can find a minimum in $\mathcal{O}(n)$ time by considering these points and the points $f(0)$ and $f(\ell)$. This solves our subproblem.

Having found local minima and maxima of f and ordered them by their ordinates, it is easy to evaluate $f(x)$ for arbitrary $x \in [0, \ell]$ in $\mathcal{O}(\log n)$ time.

Now, we wish to minimize a function $R(x) = g(x) + h(x)$, where both $g(x)$ and $h(x)$ are functions with the form of Equation (6). We note that $R(\alpha)$ as given in Equation (5) is of this form. We solve the minimization problem for $g(x)$ and $h(x)$, keeping track of the valley points for $g(x)$ and $h(x)$. The minimum value of a function that is the sum of two sawtooth functions must be at an x value that corresponds to a valley in one of the sawtooth functions or at 0 or ℓ. Hence, we need only compute the value of $R(x)$ at each of the valleys of $g(x)$ and $h(x)$. Using the binary search method to evaluate the other (non-valley) function at x, we see that evaluating $R(x)$ at any point takes $\mathcal{O}(\log n)$ time. Hence the minimum can be found by evaluating $R(x)$ at each of at most $2n$ points, in time $\mathcal{O}(n \log n)$.

Thus, solving this minimization problem for each edge then takes at most $\mathcal{O}(|V| \log |V|)$ time, and the total running time of the algorithm will then be $\mathcal{O}(|V|^3 + |E||V| \log |V|)$.

3 A Sufficient Set of SPSTs

We have investigated one way to look for an optimal solution to k-MEST. Another approach follows the idea of an algorithm for uniformly weighted graphs (ℓ constant) given in [1]. The idea is to try to generate a set of spanning trees Q such that for any point ϕ on G, some tree in Q is an SPST rooted at ϕ. It turns out that we can construct Q in polynomial time, yielding another algorithm for k-MEST. The key is that all edges in G can be partitioned into intervals such that any two points in an interval have identical sets of shortest path spanning trees. Further, there are at most $|V|+1$ such intervals on any given edge. These intervals are determined by the points γ_v defined in Equation(1). We construct the set Q by including one SPST for each vertex and one for each interval. This guarantees that some tree T in Q will be an SPST rooted in the same interval as a special point χ. Thus an optimal solution to k-MEST can be found by finding the tree T in Q that minimizes $\max_{s\in S, v\in V} d_T(s,v)$.

While this approach solves k-MEST in a less efficient manner than the previous one, it may be of more general interest. The set of trees Q can be evaluated based on any criteria, not only maximum source eccentricity. Hence, it may be useful in the solution of problems other than k-MEST.

Index the γ_v's defined by Equation (1) and the associated vertices so that

$$d_p(\gamma_{v_1}), d_p(\gamma_{v_2}), \ldots, d_p(\gamma_{v_n})$$

is a non-decreasing sequence of distances on the edge (p,q). The three theorems given below all apply only to points in (γ_q, γ_p), but since we are assuming no long edges, this interval is precisely (p,q).

Theorem 5 *For any two points α_1 and α_2 in an interval $(\gamma_{v_i}, \gamma_{v_{i+1}})$ for $1 \le i < n$, the set of SPSTs rooted at α_1 is the same as the set of SPSTs rooted at α_2.*

Proof: The intuition is that in the interval $(\gamma_{v_i}, \gamma_{v_{i+1}})$ shortest paths from any point in the interval to some v_j go either all through p or all through q. This determines the structure of the SPSTs possible in the interval: for any point in the interval, shortest paths to v_j include p for $j \le i$ and include q for $j > i$. It follows that for any two points α_1 and α_2 in the interval, any SPST rooted at α_2 is also an SPST at α_1. □

Note that the implication holds also when α_1 is one of the endpoints of the interval, i.e., some γ_v. The following theorem is easily proved by considering the cases.

Theorem 6 *Any SPST for a point $\alpha \in (\gamma_{v_{i-1}}, \gamma_{v_{i+1}})$, $1 < i < n$, is also an SPST for the point γ_{v_i}.*

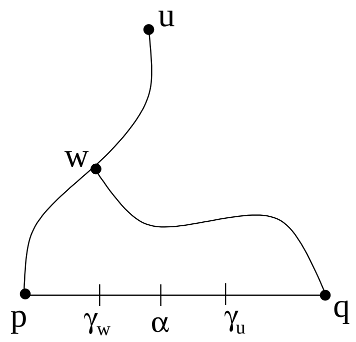

Figure 2: Possible cycle situation in T

Theorem 7 *Given points γ_v, for all $v \in V$, on an edge (p, q) along with SPSTs T_p and T_q rooted at p and q respectively, a SPST for any $\alpha \in (\gamma_q, \gamma_p)$ can be constructed by the following method. Let L be the set of edges in T_p that are on a shortest path to some v with $d_p(\alpha) \leq d_p(\gamma_v)$. Let R be the set of edges in T_q that are on a shortest path to some v with $d_p(\gamma_v) < d_p(\alpha)$. Then, the graph induced by the edges $L \cup R \cup (p, q)$ is an SPST for α.*

Proof: Let T be the graph induced by the edge set $L \cup R \cup (p, q)$. We must show that T is acyclic and contains shortest paths from α to v for each $v \in V$. Our definition of the points γ_v and the selection of edges from T_p and T_q immediately satisfies the shortest path condition. A cycle can only occur in T if there is some vertex w reached in T via both T_p and T_q. This cannot happen directly as we always chose either a path in T_p or a path in T_q to a given vertex, but not both. Thus, this situation can occur only if there is some u reached via T_p with a w on the $p \rightsquigarrow u$ path such that w is reached via q (where we have picked vertices names without loss of generality). This situation is shown in Figure 2. This scenario implies $d_p(\gamma_w) < d_p(\alpha) < d_p(\gamma_u)$. By definition of γ_w, the path $\alpha \rightsquigarrow q \rightsquigarrow w$ is shorter than the path $\alpha \rightsquigarrow p \rightsquigarrow w$ But this means that the shortest path to u is also through q, contradicting our assumption. Hence $\gamma_w \geq \gamma_u$ and T contains no cycles. $\Box$

4 Conclusions

This paper effectively solves the k-MEST problem, one of two multi-source spanning tree problems defined in [1]. We note that, in keeping with the optimization problems associated with the two cost measures discussed there, other problems can be similarly defined; among others

$$\begin{aligned}
c(T) &= \max_{s \in S}\{ \sum_{v \in V}\{d_T(s,v)\} \} \\
c(T) &= \max_{v \in V}\{ \sum_{s \in S}\{d_T(s,v)\} \} \\
c(T) &= \sum_{s \in S}(\max_{v \in V}\{d_T(s,v)\}) \\
c(T) &= \sum_{v \in V}(\max_{s \in S}\{d_T(s,v)\})
\end{aligned}$$

The method of generating a set of shortest path spanning trees given in Section 3 can be useful as part of exact or approximation algorithms for other problems, in particular the problems defined by the metrics given above.

Although our results have been presented in a purely theoretical setting, we believe that they may have some bearing on routing in networks. Indeed, the original discussion in [1] has been motivated by problems of determining good shared network substructures used in group communications. While this paper does not directly address the question of how effective optimal k-MEST trees will be for network communication, relating the described measures to the praxis of group communications and evaluation of the performance of defined by those measures optimal spanning trees would be an interesting and potentially fruitful exercise.

References

[1] Arthur M. Farley, Paraskevi Fragopoulou, David W. Krumme, Andrzej Proskurowski, and Dana Richards. Multi-source spanning tree problems. In C. Gavoile, J.C. Bermond, and A. Raspaud, editors, *Proceedings in Informatics 5 (SIROCCO 6)*, pages 126–136, 1999.