Approximation of the Constrained Path Covering Problem

Laurent Alfandari

Approximation of the Constrained Path Covering Problem

Laurent Alfandari*†
* LIPN, UMR-CNRS 7030, Université Paris XIII, France
† ESSEC,BP 05105 95021 Cergy-Pontoise, France
E-mail : *alfandari@essec.fr*

Abstract

We study a generic covering problem frequently met in transportation planning. The problem is to cover a set of tasks by constrained paths, representing routings of commodities, so that the total cost of selected paths is minimal. In this paper, paths are constrained to have limited weight and limited number of tasks. We show that the generic problem is NP-hard and that a greedy heuristic using dynamic programming at each step achieves an approximation ratio of $\ln d$, where d is the maximum number of tasks of a path.

Keywords: transportation planning, constrained path, set covering, dynamic programming, approximation.

1 Problem statement

The following generic minimization problem, that we call Constrained Path Covering Problem (CPCP) in this paper, has many applications especially for fleet scheduling or crew pairing in transportation planning [7, 4]. We are given a set of tasks $V = \{1, \ldots, n\}$ to be covered by a set K of commodities (for example, flight legs to be covered by crews or planes, trains to be covered by pilots or locomotives). With each commodity $k \in K$ is associated an acyclic graph $G^k = (V^k \cup \{s^k, t^k\}, A^k)$ where $V^k \subseteq V$ is the subset of tasks that commodity k can perform, arc $(i, j) \in A^k \cap V^2$ if and only if commodity k can perform task j after task i, $(s^k, i) \in A^k$ (resp. $(i, t^k) \in A^k$) if i starts (resp., ends) at the base of commodity k. With each arc $(i, j) \in A^k$ is associated a cost c^k_{ij} and a weight w^k_{ij}. There is also a zero-cost and zero-weight direct arc (s^k, t^k) in G^k. The problem is to find in each graph G^k a path from s^k to t^k (possibly reduced to the single arc (s^k, t^k), meaning that commodity k is not used) so that each task $i \in V$ is covered by at least one of these paths, each path contains at most d^k tasks, the weight of a path is at most b^k, and the total cost of the selected paths is minimum.
We note $\mathcal{P}^k$ the set of feasible

The general problem can formulate as the following binary Linear Program:

$$\text{Min} \quad \sum_{k\in K}\sum_{(i,j)\in A^k} c_{ij}^k x_{ij}^k \tag{1}$$

$$\text{s.t.} \quad \sum_{k\in K}\sum_{i:(i,j)\in A^k} x_{ij}^k \geq 1 \qquad \forall j \in V \tag{2}$$

$$\text{(CPCP)} \quad \sum_{j:(s^k,j)\in A^k} x_{s^k j}^k = \sum_{i:(i,t^k)\in A^k} x_{it^k}^k = 1 \quad \forall k \in K \tag{3}$$

$$\sum_{i:(i,j)\in A^k} x_{ij}^k = \sum_{l:(j,l)\in A^k} x_{jl}^k \qquad \forall k \in K, j \in V^k \tag{4}$$

$$\sum_{(i,j)\in A^k} x_{ij}^k \leq d^k - 1 \qquad \forall k \in K \tag{5}$$

$$\sum_{(i,j)\in A^k} w_{ij}^k x_{ij}^k \leq b^k \qquad \forall k \in K \tag{6}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall k \in K, (i,j) \in A^k \tag{7}$$

The objective function (1) minimizes the sum of the costs of selected paths. A special case where the objective is to minimize the number of used commodities can be handled by setting data $c_{s^k i}^k = 1$ for $(s^k, i) \in A^k$ and all other costs are zero. This criterion can also be treated in lexicographical order compared to other costs on arcs. The global constraint (2) expresses the need of covering each task of V by at least one path. Constraints (3-6) are local constraints holding for each path, which can be decomposed as a first block of structural constraints defining a $s^k - t^k$-path (3-4) and a second block of resource constraints (5-6) limiting the number of tasks and the weight of a path. The right hand side of constraint (5) is $d^k - 1$ because a path of a arcs starting from s^k and ending at t^k contains $a - 1$ tasks, i.e. nodes of V^k.
We note $\mathcal{P}^k$ the set of $s^k - t^k$ feasible paths of G^k, i.e., having weight no more than b^k and number of tasks no more than d^k.

The paper is structured as follows. Section 2 is devoted to the NP-hardness of the problem, its reformulation as an exponential-size set covering problem, the general greedy process and its worst-case performance analysis. Section 3 studies approximation of the NP-hard subproblem of the greedy process, which is shown to admit a Fully Polynomial Time Approximation Scheme (FPTAS) using dynamic programming iterations on scaled and rounded costs. Section 4 concludes the paper.

2 Complexity and approximation

Proposition 1. *CPCP is NP-hard.*

Proof. We reduce the NP-hard Set Covering Problem (SCP) to CPCP. Given a set C of elements and a collection $\mathcal{S} = \{S_1, \ldots, S_m\}$ of subsets of C with cost $c(S)$ for $S \in \mathcal{S}$, SCP consists in finding a minimum cover of C, i.e., a subset $\mathcal{S}' \subseteq \mathcal{S}$ such that $\cup_{S\in\mathcal{S}'} S = C$ and total cost $\sum_{S\in\mathcal{S}'} c(S)$ is minimum. We transform SCP instance $(C, \mathcal{S})$ into a CPCP instance the following way. We set $V = C = \{1, 2, \ldots, n\}$ and for $k = 1, \ldots, m$, we construct acyclic graph $G^k = (V^k \cup \{s^k, t^k\}, A^k)$ where

$V^k = S_k = \{i_1^k, i_2^k, \ldots, i_{|S_k|}^k\}$, $A^k = \{(s^k, t^k), (s^k, i_1^k), (i_{|S_k|}^k, t^k)\} \cup \{(i_l^k, i_{l+1}^k) : l = 1, \ldots, |S_k| - 1\}$. We set moreover $c^k_{s^k i_1^k} = c(S_k)$, all other arcs have zero-cost. Finally, we set $w_{ij}^k = 1$ for all arc $(i,j) \in A^k$, $b^k = n+1$ and $d^k = n$ for all k so that every path is feasible in G^k. It is easy to see that every cover in the SCP instance can be transformed in a CPCP solution of the same cost and vice-versa by associating subset $S_k = \{i_1^k, i_2^k, \ldots, i_{|S_k|}^k\}$ in SCP and path $(s^k, i_1^k, i_2^k, \ldots, i_{|S_k|}^k, t^k)$ in CPCP.□

The objective of polynomial approximation is to find for NP-hard problems a polynomial-time algorithm that finds a solution whose objective is always within some factor, as small as possible, of the optimal value. We show that we can reformulate CPCP as a particular exponential-size SCP, and derive from the best-known approximation for SCP a polynomial-time heuristic for CPCP achieving a logaritmic approximation ratio.

Proposition 2. *Let I be an arbitrary CPCP instance. Consider the following SCP instance $I' = (C, \mathcal{S})$ where $C = V$, $\mathcal{S} = \cup_{k \in K}\{V(p) : p$ is a feasible path in $G^k\}$, and for $S \in \mathcal{S}$, $c(S)$ is the cost of the corresponding path in G^k. Then every CPCP-solution of I can be transformed in a SCP-cover of I' of the same cost, and vice-versa.*

Proof. Every CPCP feasible solution $\{p^1, \ldots, p^{|K|}\}$ for instance I, where p^k is the selected path in subgraph G^k, is in 1-1 correspondence with feasible cover $\{V(p^1), \ldots, V(p^{|K|})\}$ in I' and these solutions have obvioulsy the same cost. □

We deduce from proposition 2 that solving the SCP reformulation of CPCP is equivalent to solving the original problem. The best approximation algorithm for SCP is the classical *greedy* algorithm analysed by Chvátal [3]. It starts with $U = C$ (U represents the current set of uncovered elements of C), repeatedly solves a *subproblem*, which consists in picking at each step a subset $S^* \in \mathcal{S}$ so that

$$\frac{c(S^*)}{|S^* \cap U|} = \min_{S \in \mathcal{S}} \frac{c(S)}{|S \cap U|} \tag{8}$$

and sets $U \leftarrow U \setminus \{S^*\}$, until all elements of C are covered, i.e. $U = \emptyset$. This *greedy* process achieves an approximation ratio of $H(\sigma)$, where $\sigma = \max_{S \in \mathcal{S}} |S|$ and $H(\sigma) = \sum_{1 \leq i \leq \sigma} 1/i \leq \ln \sigma + 1$ [3]. Although CPCP reduces to a particular SCP by proposition 2, it is not obvious whether the above *greedy* algorithm can run on the Set Covering reformulation of CPCP, as the number of subsets $|\mathcal{S}|$ is exponential at worst case ($|\mathcal{P}^k| = O(2^{|V^k|}) = O(2^n)$) so the computation of the *subproblem* ratio (8) might be tricky. Nevertheless, we can show as in [1] that complete enumeration of these subsets is not needed so that the *subproblem* remains tractable. In this paper, as in [2], the subproblem is NP-hard but can be approximated by a fully polynomial-time approximation scheme (FPTAS), which enables to conserve the logaritmic approximation ratio of *Greedy* for the original master problem.

Proposition 3. [1, 2] *Consider an instance $(C, \mathcal{S})$ of the Set Covering problem. If subproblem (8) can be approximated within ratio $1+\epsilon$, then the associated greedy heuristic approximates the Set Covering instance within ratio $(1+\epsilon)H(\sigma)$, where $\sigma = \max_{S\in\mathcal{S}} |S|$.*

The proof is a direct application of Chvátal's result [3]. For CPCP, adapting the above greedy heuristic we start with $U = V$. At each step of the greedy process, there are exactly $|K|$ subproblems and each subproblem (SP^k) is defined as follows.

Definition 1. *Given $U \subseteq V$, the CPCP subproblem (SP^k) for $k \in K$ consists in finding in graph G^k a path minimizing ratio $c(p)/|V(p)\cap U|$ over all paths $p \in \mathcal{P}^k$.*

We show in next section that subproblem (SP^k) admits a FPTAS providing a feasible path $p^k \in \mathcal{P}^k$ whose objective $c(p^k)/|V(p^k) \cap U|$ lies within ratio $(1+\epsilon)$ of the optimum. Then the global best path $p^* = arg\min\{c(p^k)/|V(p^k)\cap U| : k \in K\}$ is added to the greedy solution, the set U of uncovered tasks is set to $U\setminus V(p^*)$, and the greedy process is iterated again until $U = \emptyset$. By propositions 2 and 3, the worst-case ratio of this greedy process is bounded above by $1 + \ln(d)$ where $d = \max_{k\in K} d^k$. It remains to show then that subproblem (SP^k) admits a Fully-Polynomial Time Approximation Scheme.

3 Approximation of the CPCP subproblem

Observation. *In order to keep notations simple while studying a subproblem (SP^k) of definition 1, for the whole section we drop the k-index in graph G^k and in cost and weight data. Hence we have a graph $G = (\{s,t\} \cup V, A)$, with costs c_{ij} and weights w_{ij}, and the subproblem (SP) is to find in G a path $p \in \mathcal{P}$ minimizing ratio $c(p)/|V(p) \cap U|$, where $\mathcal{P}$ is the set of all $s-t$ paths p of G satisfying $|V(p)| \leq d$ and $w(p) \leq b$. We note $OPT = \min_{p\in\mathcal{P}} c(p)/|V(p) \cap U|$.*

Let $u_j = 1$ if $j \in U$, $u_j = 0$ otherwise. Let us first consider the two-step Dynamic Programming (DP) algorithm 1 which, given a cost vector $\hat{c}$ on arcs and an upper bound B:

(i) finds at step 1, for all node $j \in V \cup \{t\}$, the minimum weight $w_j^*(q)$ of a path among all $s-j$ paths p satisfying $|V(p)\cap U| = q$, for $q \leq d$, and keeps in a set Q_j only those feasible values of q, i.e., such that $w_j^*(q) \leq b$,

(ii) finds at step 2, for all $q \in Q_t$, the minimum cost $\hat{c}_q^*$ of a $s-t$ path among all feasible paths $p_q \in \mathcal{P}$ satisfying $|V(p_q) \cap U| = q$ and $\hat{c}(p_q)/q \leq B$.

The complexity of step 1, as all nodes and for each node all predecessors are examined, is in $O(|A|)$. At step 2, for each iteration of the while loop all nodes $j \in V \cup \{t\}$, all predecessors of j and all values of Q_j are enumerated, so the complexity of one iteration is in $O(|A|d)$ as $\max_j |Q_j| \leq d$. The overall complexity of DP is derived then in lemma 1.

Algorithm 1 : DP$(\hat{c}, B)$
Begin

Step 1 `// outputs for each` $j \in V$ `the set` $Q_j = \{q \in \{0, \ldots, d\} : w_j^*(q) \leq b\}$
$Q_s = \{0\}$
$S = \{s\}$
$\bar{S} = V \cup \{t\}$
While $\bar{S} \neq \emptyset$:
    For $j \in \bar{S}$ such that $P(j) \subseteq S$:
        $Q_j = \{q + u_j : q \in \cup_{i \in P(j)} Q_i\} \setminus \{d+1\}$
        For $q \in Q_j$:
            $w_j^*(q) = \min_{i \in P(j)}\{w_i^*(q - u_j) + w_{ij}\}$
            If $w_j^*(q) > b$ **then** $Q_j \leftarrow Q_j \setminus \{q\}$
        End for
        Output Q_j for $j \in V \cup \{t\}$
        $S \leftarrow S \cup \{j\}$; $\bar{S} \leftarrow \bar{S} \setminus \{j\}$
    End for
End while

Step 2
$c = 0$
$Q'_t \leftarrow Q_t$
While $Q'_t \neq \emptyset$:
    $c \leftarrow c + 1$
    For $j \in V$, **for** $q \in Q_j$:
        $w_j^*(c, q) = \min(\min_{i \in P(j): \hat{c}_{ij} \leq c}\{w_i^*(c - \hat{c}_{ij}, q - u_j) + w_{ij}\}; w_j^*(c-1, q))$
    For $q \in Q'_t$:
        $w_t^*(c, q) = \min(\min_{i \in P(t): \hat{c}_{ij} \leq c}\{w_i^*(c - \hat{c}_{ij}, q - u_j) + w_{ij}\}; w_t^*(c-1, q))$
    End for
    For $q \in Q'_t$:
        if $w_t^*(c, q) \leq b$ **then** $\hat{c}_q^* = c$; $Q'_t \leftarrow Q'_t \setminus \{q\}$
        if $c \geq qB$ **then** $\hat{c}_q^* = \infty$; $Q'_t \leftarrow Q'_t \setminus \{q\}$
    End for
End while
End.

Lemma 1. *Given cost vector $\hat{c}$ and upper bound B, algorithm $DP(\hat{c}, B)$ runs in time $O(|A|d\min(dB, \max_{q\in Q_t} \hat{c}^*_q))$.*

For solving exactly the subproblem, we could set cost vector $\hat{c} = c$, upper bound $B = \infty$ and simply output $OPT = \min\{\hat{c}^*_q/q : q \in Q_t\}$. In that case however, the value $\max_{q\in Q_t} \hat{c}^*_q$ may be arbitrary large and the complexity of the DP procedure would not be polynomial in the data (which of course is consistent with the fact that the subproblem is NP-hard). We thus use a technique inspired from [5] (this paper being itself inspired from [6]) for approximating OPT by a Fully Polynomial Time Approximation Scheme. The main idea is to start with a lower bound LB and an upper bound UB on OPT; then, an iterative reduction of interval $[LB, UB]$ is performed by testing at each iteration whether $OPT \geq R$ or $OPT \leq (1+\delta)R$, where R is an appropriate value inside interval $[LB, UB]$, and updating bounds in both cases. This test is correctly done running algorithm $\mathrm{DP}(\hat{c}, d/\delta)$, where $\hat{c}_{ij} = \lfloor c_{ij}/(R\delta/d)\rfloor$, as shown in Lemma 2. When the ratio UB/LB falls under a prefixed constant, the DP procedure is applied on scaled and rounded costs in order to achieve the $(1+\epsilon)$ approximation claimed for the subproblem.

Lemma 2. *Let $R \in [LB, UB]$, $0 < \delta < d$ and $\hat{c}_{ij} = \lfloor c_{ij}/(R\delta/d)\rfloor$ for all $(i, j) \in A$ such that $c_{ij} \leq R$, else $\hat{c}_{ij} = \infty$. At the end of procedure $DP(\hat{c}, d/\delta)$, which runs in time $O(|A|d^3/\delta)$, if $\min_{q\in Q_t} \hat{c}^*_q/q \leq d/\delta$ then $OPT \leq R(1+\delta)$ else $OPT \geq R$.*

Proof. The complexity of $DP(\hat{c}, d/\delta)$ directly follows from Lemma 2. Now, if for some $q \in Q_t$ we have $\hat{c}^*_q/q \leq d/\delta$ then there is a feasible path $p_q \in \mathcal{P}$ satisfying $|V(p) \cap U| = q$ and $\hat{c}(p_q)/q \leq d/\delta$, and then

$$\begin{aligned}
OPT \leq \frac{c(p_q)}{q} &= \frac{1}{q}\sum_{(i,j)\in p_q} c_{ij} \\
&\leq \frac{1}{q}\sum_{(i,j)\in p_q}\left(\left\lfloor\frac{c_{ij}}{R\delta/d}\right\rfloor + 1\right)\frac{R\delta}{d} \qquad \text{as } x/y \leq \lfloor x/y\rfloor + 1 \\
&= \frac{1}{q}\left(\frac{R\delta}{d}\left(\hat{c}(p_q) + |p_q|\right)\right) \\
&\leq (R + R\delta/q) \qquad \text{as } \hat{c}(p_q)/q \leq d/\delta \text{ and } |p_q| \leq d \\
&\leq (1+\delta)R \text{ as } q \geq 1
\end{aligned}$$

On the other side, if for all $q \in Q_t$ $\hat{c}^*/q > d/\delta$ then if p^* is an optimal path of value OPT, we have $\hat{c}(p^*)/|V(p^*)\cap U| > d/\delta$ and then $OPT = c(p^*)/|V(p^*)\cap U| > \hat{c}(p^*)(R\delta/d)/|V(p^*)\cap U| \geq R$ as $\hat{c}(p)/|V(p)\cap U| \geq d/\delta$ for all $p \in \mathcal{P}$. □

We are now ready to introduce the algorithm. Let p^*_c and p^*_w denote respectively a minimum-cost $s-t$ path, and a minimum-weight $s-t$ path among paths with no more than d nodes of $V \setminus U$ (the latter can be computed by dynamic programming, not extending labels with more than d nodes outside U). We set the initial Lower

Bound LB_1 to $c(p_c^*)/d$ and the initial Upper Bound UB_1 to $c(p_w^*)/|V(p_w^*) \cap U|$. Finally, $\mu > 1$ and $\beta \in]0;1[$ are constant parameters (in [5], $\mu = 2$ and $\beta = 1/2$).

Algorithm 2 : approximation scheme for (SP)
Begin
$r = 1$;
While $UB_r > \mu LB_r$:
$\delta_r \leftarrow (UB_r/LB_r)^{1-\beta} - 1$
$R_r \leftarrow UB_r^{\beta/2} LB_r^{1-\beta/2}$
$c_{ij}^r \leftarrow \lceil c_{ij}/(R_r\delta_r/d) \rceil$ for all $(i,j) \in A$ s.t. $c_{ij} \leq R_r$, else $c_{ij}^r \leftarrow 0$
run DP$(c^r, d/\delta_r)$
If $\min_{q \in Q_t} \hat{c}_q^* \leq d/\delta_r$ **then** $UB_{r+1} \leftarrow (1+\delta_r)R_r$ **else** $LB_{r+1} \leftarrow R_r$
End while
Obtain an $(1+\epsilon)$-approximation of OPT by running $DP(\lfloor c/(LB\epsilon/d) \rfloor, d)$.
End.

The complexity of r^{th} run $DP(c^r, d/\delta_r)$, following Lemma 1, is $O(|A|d^3/\delta_r)$. We rewrite for general parameters μ and β the proof of [5] , showing that $\sum_{1 \leq r \leq l} 1/\delta_r = O(1)$, where l is the total number of runs of the while loop. Then the whole complexity of solving one subproblem is $O(|A|d^3)$.
If at the end of DP the lower bound is modified as $LB_{r+1} = R_r$ then $UB_{r+1}/LB_{r+1} = UB_r/(UB_r^{\beta/2} LB_r^{1-\beta/2}) = (UB_r/LB_r)^{1-\beta/2}$. Otherwise, if $UB_{r+1} = (1+\delta_r)R_r$ then $UB_{r+1}/LB_{r+1} = (1+\delta_r)R_r/LB_r = (UB_r/LB_r)^{1-\beta}(UB_r^{\beta/2} LB_r^{1-\beta/2})/LB_r = (UB_r/LB_r)^{1-\beta/2}$. Hence in both cases,

$$UB_{r+1}/LB_{r+1} = (UB_r/LB_r)^{1-\beta/2}$$

As $1/\delta_r = 1/((UB_r/LB_r{}^{1-\beta}) - 1)$ and $UB_r/LB_r > \mu$ we have

$$(LB_r/UB_r)^{1-\beta} \leq 1/\delta_r \leq \frac{1}{1-1/\mu^{1-\beta}} (LB_r/UB_r)^{1-\beta}$$

and then $\sum_{1 \leq r \leq l} 1/\delta_r = O(\sum_{1 \leq r \leq l} (LB_r/UB_r)^{1-\beta})$. We have

$$\begin{aligned}
\sum_{r=1}^{l} (LB_r/UB_r)^{1-\beta} &= \sum_{r=1}^{l} (LB_l/UB_l)^{(1-\beta).(2/(2-\beta))^{l-r}} \\
&= \sum_{j=0}^{l-1} (LB_l/UB_l)^{(1-\beta).(2/(2-\beta))^{j}} \\
&\leq \sum_{j=0}^{l-1} \mu^{-(1-\beta).(2/(2-\beta))^{j}} \\
&\leq \mu^{-(1-\beta)} \sum_{j=0}^{l-1} \left(\mu^{-(1-\beta)\frac{2}{(2-\beta)}-1}\right)^{j} \\
&\leq \mu^{-(1-\beta)}/(1-\mu^{-(1-\beta)\frac{2}{(2-\beta)}-1})
\end{aligned}$$

So $\sum_{1\leq r\leq l} 1/\delta_r \leq \left(\frac{1}{1-1/\mu^{1-\beta}}\right)\left(\mu^{-(1-\beta)}/(1-\mu^{-(1-\beta)\frac{2}{(2-\beta)}-1})\right)$ which is a function of two constants μ and β, hence $O(1)$. $\square$

We thus conclude that subproblem (SP) admits a FPTAS running in $O(|A|d^3)$. Going back to CPCP, each of the $|K|$ subproblems is approximated in time $O(|A^k|(d^k)^3)$. Hence the whole greedy heuristic described in section 2 for CPCP runs in time $O(|V||K||A|d^3)$, achieving an approximation ratio of $1+\ln d$ where $d=\max_k d^k$.

4 Conclusion

We presented a polynomial-time greedy heuristic for the minimum Constrained Path Covering Problem providing a feasible solution with cost at most $1+\ln d$ times the optimal cost, which is particularly interesting for small values of the maximum allowed number of tasks on a path. In real-case applications, this parameter is often limited indeed as for example when paths represent plannings of crews the number of tasks they can perform within their duty period is generally restricted by social rules of the company. Let us note finally that the approximation result holds because the min-ratio subproblem is submitted to a single (knapsack) resource constraint. It would not hold anymore for more resource constraints, no approximation schemes generally exist in that latter case.

References

[1] Alfandari L., V. Paschos. Master-slave strategy and polynomial approximation. *Computational optimization and Applications* 16:3 (2000) 231-245.

[2] Alfandari L. Improved approximation of the Soft-Capacitated Facility Location Problem. *RAIRO Operations Research* 41 (2007) 43-93.

[3] Chvátal V. A greedy heuristic for the set covering problem. *Mathematics of Operations Research* 4:3 (1979) 233-235.

[4] Desaulniers G., J. Desrosiers, Y. Dumas, S. Marc, B. Rioux, MM. Solomon, F. Soumis. Crew pairing at Air France. *European journal of operational research* 97:22 (1997) 245-259.

[5] Ergun F., R. Sinha and L. Zhang. An Improved FPTAS for Restricted Shortest Path. *Information Processing Letters* 83:5 (2002) 237-293.

[6] Hassin R. Approximation schemes for the restricted shortest path problems. *Mathematics of Operations Research* 17:1 (1992) 36-42.

[7] Ziarati K., F. Soumis, J. Desrosiers, M.M. Solomon. A branch-first cut-second approach for locomotive assignment. *Management Science* 45:8 (1999) 1156-1168.