Institute for Logic, Language and Computation

# A NOTE ON THE COMPLEXITY OF LOCAL SEARCH PROBLEMS

Sophie Fischer

ILLC Prepublication Series
for Computation and Complexity Theory CT-93-02

University of Amsterdam

**Institute for Logic, Language and Computation**
Plantage Muidergracht 24
1018TV Amsterdam
Telephone 020-525.6051, Fax: 020-525.5101

# A NOTE ON THE COMPLEXITY OF LOCAL SEARCH PROBLEMS

Sophie Fischer
Department of Mathematics and Computer Science
University of Amsterdam

*ILLC Prepublications*
*for Computation and Complexity Theory*
ISSN 0928-3323

*Coordinating editor: Dick de Jongh*

Research supported by
the Netherlands Organization for Scientific Research (NWO)

received March 1993

# A note on the complexity of local search problems

Sophie Fischer

**Abstract**

In this paper, we study the complexity of finding a local optimum in combinatorial optimization problems. For many optimization problems derived from NP-complete decision problems, e.g. MAXCLIQUE, a locally optimal solution starting from a given solution can be found in (deterministic) polynomial time. It is not known however, whether this is a property shared by all these problems. In particular the class of PLS-complete problems seems to form an exception. In this paper we present several problems for which the question, whether for a given solution a local optimum *exists* within a polynomial number of PLS steps, is NP-complete.

## 1 Introduction

Combinatorial optimization problems form a central object of study in operations research. Unfortunately, as for most of these problems the underlying decision problems are NP-complete, there is very little hope of finding feasible algorithms that produce optimal solutions (unless, of course, P=NP). Proving a problem to be difficult does not make the problem go away. Research interest has therefore shifted from hunting after optimal solutions, to directions in which near optimal solutions were expected to be found. A very recent result by Arora et al. [ALM+92] shows that we cannot expect too much in this direction also. In particular, they proved for several problems, that a global optimum cannot be approached within a constant factor, unless P=NP. A third approach for attacking the combinatorial optimization problems is to use local search strategies. Instead of searching for (an approximation of) a global optimum, a given solution is improved upon by polynomial time computable changes, until a *local* optimum is reached. For several optimization problems with NP-complete decision variants, e.g. CLIQUE, the question of producing a locally optimal solution from a given solution can easily be seen to be polynomial time solvable. Johnson, Papadimitriou and Yannakakis [JPY88] defined a class of optimization problems PLS that are sensitive to such an attack. This class of problems has its own type of reductions (PLS-reductions), and this type of reductions gives rise to the identification of complete problems in the class PLS. We give a definition of a PLS-reduction in the next section, but the idea is that a PLS reduction allows the transformation of an instance of a problem $A$ into an instance of a problem $B$ with the property that a locally optimal solution found for problem $B$ can be translated back to a locally optimal solution for problem $A$. All transformations are, of course, polynomial time bounded. In this way, computing a locally optimal solution for $A$ from a given solution cannot be harder than computing a locally optimal solution from a given solution for $B$.

It is not at all clear which problems in PLS permit strategies of finding a local optimum from a given solution in polynomial time. In particular, for the class of PLS-complete problems this question is open to date. To obtain insight in the difficulty of finding locally optimal solutions, Papadimitriou, Schäffer and Yannakakis [PSY90] considered the following problem. Given a local search problem, a start solution $s$, and a locally optimal solution $s'$, *how hard is it* to decide whether $s'$ is reachable from $s$? In the same paper they proved that for some PLS problem this question is PSPACE-hard.

Deciding reachability of a given solution does not have to state anything about feasibility of a local search approach for the problem. An optimum may be reachable, yet any local search algorithm working up to that solution may have to spend an exponential number of steps. We feel therefore, that the question whether a feasible strategy exists is better formulated by the question: Given an instance of a local search problem, and a start solution $s$; does there exist a locally optimal solution $s'$ that can be reached from $s$ within a polynomial number of PLS steps? This question cannot be PSPACE-complete unless NP=PSPACE, since it is easily seen to be in NP. However the question may still be difficult to answer. We prove for several problems in PLS that this question is NP-complete. On the other hand there are also problems in PLS, for which this question is easy to answer. We prove for several PLS-complete problems that this question is in P.

## 2 Definitions and Notations

In this section we give definitions and notations, used in this paper. We use $|\alpha|$ to denote the number of bits in string $\alpha$.

**Definition 2.1** *Let $\alpha$ and $\beta$ be two binary strings of length $n$. The* hamming distance *between $\alpha$ and $\beta$, denoted by $\mathcal{HD}(\alpha, \beta)$, is the number of bits in which $\alpha$ and $\beta$ differ.*

**Definition 2.2** *An NP-optimization problem $A$, is a four-tuple, $A = \langle I_A, \mathcal{FS}_A, f_A, opt_A \rangle$, where*

- *$I_A$ is the set of instances of $A$. It is assumed that $I_A$ is recognizable in polynomial time.*
- *$\mathcal{FS}_A : I_A \rightarrow 2^{\{0,1\}^*}$, assigns to every instance $I \in I_A$ a set of feasible solutions of $I$. There must be a polynomial $q$ and a polynomial time computable predicate $\pi$, such that $\forall I \in I_A$, $\mathcal{FS}_A(I) = \{s; |s| \leq q(|I|) \wedge \pi(I, s)\}$. The polynomial $q$ and the predicate $\pi$ only depend on $A$.*
- *$f_A : I_A \times \{0,1\}^* \rightarrow N_0$ assigns to every $s \in \mathcal{FS}_A(I)$ an integer value. This integer value is the* cost *of feasible solution $s$. If $s \notin \mathcal{FS}_A(I)$, then $f_A(I, s)$ is undefined. The function $f_A$ can be computed in polynomial time.*

- $opt_A \in \{max, min\}$, *is used to indicate whether A is a minimization or a maximization problem.*

A special class of NP-optimization problems is formed by the *polynomially bounded* NP-optimization problems.

**Definition 2.3** *Let A be a NP-optimization problem. Let* $opt_A(I) = opt\{f_A(I,s) | s \in \mathcal{FS}_A(I)\}$. *The NP-optimization problem A is* polynomially bounded *if there is a polynomial p, such that* $opt_A(I) \leq p(|I|)$, $\forall I \in I_A$.

In case of computing optimal solutions for a NP-optimization problem is not feasible, it is sometimes possible to compute near optimal solutions.

**Definition 2.4** *Let* $A = \langle I_A, \mathcal{FS}_A, f_A, min\rangle$ *be a NP-minimization problem. Algorithm* $\mathcal{A}$ *approximates A in polynomial time within constant* $\kappa$, *if* $\forall I \in I_A$, $\mathcal{A}(I) \in \mathcal{FS}_A(I)$ *and* $\left|\frac{f_A(I,\mathcal{A}(I))}{f_A(I,s^*)}\right| \leq \kappa$, *where* $s^* \in \mathcal{FS}_A(I)$, *such that* $f_A(I,s^*)$ *is minimal, and the running time of* $\mathcal{A}(I)$ *is bounded by* $p_{\mathcal{A}}(|I|)$, $p_{\mathcal{A}}$ *a polynomial.*

The same kind of definition can be given for a NP-maximization problem.
The complexity class PLS is the class of polynomial local search problems.

**Definition 2.5** *The class PLS contains the problems* $A = \langle I_A, \mathcal{FS}_A, f_A, opt_A, N_A\rangle$, *where*

- $\langle I_A, \mathcal{FS}_A, f_A, opt_A\rangle$ *defines a NP-optimization problem with an extra condition. It is required that* $\forall I \in I_A$ *an initial solution* $s_0 \in \mathcal{FS}_A(I)$ *can be computed in polynomial time.*

- $N_A : I_A \times \{0,1\}^* \rightarrow 2^{\{0,1\}^*}$ *assigns to every* $s \in \mathcal{FS}_A(I)$ *a set of feasible solutions* $S \subset \mathcal{FS}_A(I)$. *Set S is called the set of neighbors of s, and satisfies*

  1. $\forall s \in \mathcal{FS}_A(I)$ *it can be decided in polynomial time whether s is locally optimal, i.e. whether s has a better cost than all* $s' \in N_A(I,s)$.
  2. $\forall s \in \mathcal{FS}_A(I)$, *if s is not locally optimal, a solution* $s' \in N_A(I,s)$ *with a better cost than s can be computed in polynomial time.*

*If* $s \notin \mathcal{FS}_A(I)$, *then* $N_A(I,s)$ *is undefined.*

It is not necessarily true that every NP-optimization problem satisfies the requirement that for every $I \in I_A$, an initial solution $s_0 \in \mathcal{FS}_A(I)$ can be computed in polynomial time. It is also not always true, that all neighbors of $s$ can be enumerated in polynomial time. It is even possible, that $s$ has more than a polynomial number of neighbors.
The solutions of a feasible solution set, together with a neighborhood structure, can be interpreted as a directed graph.

**Definition 2.6** *Let $A = \langle I_A, \mathcal{FS}_A, f_A, opt_A, N_A \rangle \in PLS$. For $I \in I_A$, define the* local search graph *$G_A = (V_A, E_A)$ as follows:*

$$V_A = \{s; s \in \mathcal{FS}_A(I)\}$$
$$E_A = \{(s, s'); s' \in N_A(I, s) \text{ and } f_A(I, s') > f_A(I, s)\}$$

*The edge $(s, s')$ is directed from $s$ to $s'$.*
*Let $s, s' \in \mathcal{FS}_A(I)$. Feasible solution $s'$ is* reachable *from feasible solution $s$, if there is a directed path from $s$ to $s'$ in $G_A(I)$. Such a directed path is called an* augmenting *path.*

Let $A = \langle I_A, \mathcal{FS}_A, f_A, opt_A, N_A \rangle$ be a problem in PLS. A local search algorithm, given $I \in I_A$, will first compute an initial solution $s \in \mathcal{FS}_A(I)$. This can be done in polynomial time. Then the following step is repeated, until a locally optimal solution is found.

> *The local search algorithm decides in polynomial time, whether $s$ is locally optimal. If $s$ is not locally optimal, it computes a solution $s' \in N_A(I, s)$ with a better cost than $s$. The step is repeated with $s = s'$.*

The local search algorithm will only go from one feasible solution to another with a strictly better cost. Consider the local search graph $G_A$. The local search algorithm walks along the paths of this graph. One arc in $G_A$ denotes one step of the local search algorithm.

Johnson, Papadimitriou and Yannakakis were interested in the complexity of the following problem.

> **Given** Instance $I$ of problem $A \in PLS$
> **Question** Compute a locally optimal solution in $\mathcal{FS}_A(I)$

To get a better insight in the complexity of this problem, they introduced the PLS-reduction.

**Definition 2.7** *Let $A, B \in PLS$. A PLS-reduction from $A$ to $B$ is a tuple $\langle f, g \rangle$, such that*

- *$f$ and $g$ are polynomial computable*
- *$f : I_A \rightarrow I_B$*
- *$g$ maps locally optimal solutions in $\mathcal{FS}_B(f(I))$ to locally optimal solutions in $\mathcal{FS}_A(I)$.*

*If $A$ PLS-reduces to $B$, we write $A \leq^{PLS} B$.*

# 3 Estimating the distance to a locally optimal solution

Let $A \in PLS$, $s \in \mathcal{FS}_A(I)$, $I \in I_A$. How difficult is it to decide whether a locally optimal solution can be reached from $s$, using only a polynomial number of local search steps? To investigate the complexity of this problem, we define for every $A \in$ PLS the following problem $A^*$.

**Given** $I \in I_A$, $s \in \mathcal{FS}_A(I)$, $0^d$, where $d \in Z^+$

**Question** Is there a path $p$ and a locally optimal solution $s'$, such that $p$ is an augmenting path between $s$ and $s'$ and $p$ has at most $d$ intermediate vertices

The problem $A^*$ is called the starred version of problem $A$.

We will determine now for three problems $A$ in PLS, the complexity of $A^*$. The first problem we consider is polynomially bounded. Its starred version is NP-complete. The second problem we consider is PLS-complete and its starred version is NP-complete. The third problem we consider is also PLS-complete, but for every solution we can determine in polynomial time the distance to its nearest locally optimal solution.

**Definition 3.1** $U = \langle I_U, \mathcal{FS}_U, f_U, opt_U, N_U \rangle$, *where*

- $I_U = \langle M, x, p(|x|) \rangle$, *with* $M$ *a non-deterministic Turing machine with its running time bounded by* $p(|x|)$ *on input* $x$.
- $\mathcal{FS}(\langle M, x, p(|x|) \rangle) = \{(c, t); c$ *configuration of* $M, 0 \leq t \leq 2p(|x|)$ $\}$
- $f_U(\langle c, t \rangle) = t$
- $opt_U = max$
- $N_U(\langle M, x, p(|x|) \rangle, (c, t)) = \{(c', t+1); t+1 \leq T$ *and either* $c'$ *can be reached from* $c$, *using one step of* $M$, *or* $c$ *is a rejecting final configuration and* $c = c'\}$

It is easy to see that $U \in$ PLS. It is also easy to see that $U$ is polynomial bounded.

**Theorem 3.1** $U^*$ *is NP-complete*

**Proof**

Let $A \in$ NP, $M_A$ a non-deterministic polynomial bounded Turing machine deciding $A$. Assume that $M_A$ has a running time bounded by $p_A(|x|)$ on input $x$.

Consider a function $f$, such that $f(x) = (\langle M_A, x, p_A(|x|) \rangle, \langle c_0, 1 \rangle, p_A(|x|))$, where $c_0$ is the initial configuration of $M_A$ on $x$. This function $f$ is a many-one reduction from $A$ to $U^*$.

To see this, note that $f$ can be computed in polynomial time. Furthermore, if $x \in A$, then $M_A$ can reach an accepting configuration $c_a$ within $p_A(|x|)$ steps. So from $\langle c_0, 1 \rangle$, a solution $\langle c_a, t \rangle$ is reachable, where $c_a$ is an accepting final configuration of $M_A$, and $t \leq p_A(|x|)$. The solution $\langle c_a, t \rangle$ is locally optimal.

If $x \notin A$, then all locally optimal solutions reachable from $\langle c_0, 1 \rangle$ are of the form $\langle c_f, 2p_A(|x|) \rangle$, where $c_f$ is a rejecting final configuration of $M_A$.

Therefore no locally optimal solution can be reached from $\langle c_0, 1 \rangle$, with a path of length less than or equal to $p_A(|x|)$. $\square$

The problem *CircuitFlip*, was the first problem proven to be PLS-complete, [JPY88]. It is defined as follows.

**Definition 3.2** *CircuitFlip*= $\langle I_{CF}, \mathcal{FS}_{CF}, f_{CF}, opt_{CF}, N_{CF}\rangle$, *where*

- $I_{CF} = \{C;\ C \text{ is a Boolean circuit}\}$
- $\mathcal{FS}_{CF}(C) = \{\ I{=}i_1 i_2 \dots i_n;\ I \text{ is an input for } C\}$
- $f_{CF}(C, I) = \sum_{i=1}^{m} 2^i y_i$, *where* $y_m y_{m-1} \dots y_1$ *is the output of* $C$ *on input* $I$
- $opt_{CF} = max$
- $N_{CF}(C, i_1 \dots i_j \dots i_n) = \{I'; I' = i_1 \dots \bar{i_j} \dots i_n, 1 \leq j \leq n\}$, *where* $\bar{i_j}$ *is the negation of* $i_j$*. This neighborhood is called the Flip neighborhood.*

In the *CircuitFlip* problem an input $I$ for circuit $C$ is sought, such that flipping a bit of $I$ does not improve the output of $C$. The question, stated at the beginning of this section, for *CircuitFlip* is NP-complete.

**Theorem 3.2** *CircuitFlip* is NP-complete.*

In the appendix we give a reduction $g$ from $A$ to *CircuitFlip**, $\forall A \in NP$. Thus we proof *CircuitFlip** NP-complete. Here we only give a sketch of the proof.
Let $A \in NP$, $M_A$ a non-deterministic Turing machine recognizing $A$, and $p_A(|x|)$ a bound on the running time of $M_A$ on $x$, $p$ a polynomial. Note that $p_A(|x|)$ is also a bound on the length of a configuration of $M_A$ on $x$. Let $T$ be an integer. Reduction $g$ computes on input $x$ of $A$ a Boolean circuit $C_A$, an initial solution and a distance. We assume, that every configuration of $M_A$ is followed by exactly two, not necessarily different configurations. Every configuration is time stamped, with a time stamp between 1 and $T$. With every configuration $c$ and time stamp $t$ an input of $C_A$ is associated. Suppose $M_A$ can go in one step from a configuration $c$ to a configuration $c'$. For every time stamp $t$, $1 \leq t < T$, there is a local search path in $\mathcal{FS}_{CF}(C_A)$ from an input of $C_A$ associated with $c$ and time stamp $t$ to an input of $C_A$ associated with $c'$ and time stamp $t+1$. Furthermore, for time stamp $t < T$ and rejecting configuration $c_r$, there is a local search path from an input of $C_A$ associated with $c_r$ and time stamp $t$ to an input of $C_A$ associated with $c_r$ and time stamp $t+1$.
Let $c_0$ be the initial configuration of $M_A$ on $x$, $w$ the input of $C_A$ associated with $c_0$ and time stamp 1. Let $p_1$ be a local search path from $w$ to an input of $C_A$ associated with an accepting configuration. Let $p_2$ be a local search path from $w$ to an input associated with a rejecting final configuration and time stamp $T$. Integer $T$ is chosen large enough, that $p_1$ is distinctively shorter than $p_2$.

**Corollary 1** *Let* $C$ *be an instance for CircuitFlip,* $s \in \mathcal{FS}_{CF}(C)$ *and* $\kappa$ *a constant. There is no polynomial time algorithm that approximates the distance from* $s$ *to the nearest locally optimal solution reachable from* $s$ *within constant* $\kappa$*, unless P=NP.*

### Proof

Let $C$ be a Boolean circuit, $s \in \mathcal{FS}_{CF}(C)$. Suppose $p^*$ is an augmenting (sub)path in $G_{CF}(C)$ from $s$ to $s^* \in \mathcal{FS}_{CF}(C)$, with $s^*$ locally optimal, such that the length of $p^*$ is minimal. Let $\mathcal{A}$ be an algorithm, that on input $(C, s)$ computes a local optimal solution $s' \in \mathcal{FS}_{CF}(C)$ and an augmenting (sub)path in $G_{CF}(C)$, such that $\left|\frac{length(p')}{length(p^*)}\right| \leq \kappa$. Consider the reduction from $A \in NP$ to $CircuitFlip^*$ as described above. Notice that $\kappa(3\alpha(|x|) + 6)p_A(|x|) < T$ for $|x|$ large enough. Therefore, $\mathcal{A}$ can be used to decide in polynomial time, whether $x \in A$. □

There are also PLS-complete problems $\tilde{A}$, for which it can be decided in polynomial time, whether there is a locally optimal solution $s'$ near a given solution $s$.

**Definition 3.3** *Let $A \in PLS$, $A = \langle I_A, \mathcal{FS}_A, f_A, max, N_A\rangle$. Let $\forall I \in I_A, \forall s \in \mathcal{FS}_A(I)$, $s_{max} \in N_A(I,s)$, such that $s_{max} \geq max\{f_A(I,s')|s' \in N_A(I,s)\}$. Let $\forall I \in I_A$, $T > f_A(I,s)$, for all $s \in \mathcal{FS}_A(I)$.*
*Define $\tilde{A} = \langle I_{\tilde{A}}, \mathcal{FS}_{\tilde{A}}, f_{\tilde{A}}, opt_{\tilde{A}}, N_{\tilde{A}}\rangle$, where*

- $I_{\tilde{A}} = I_A$
- $\mathcal{FS}_{\tilde{A}}(I) = \{\langle s,t\rangle;\ s \in \mathcal{FS}_A(I)$ *and* $f_A(I,s) \leq t < f_A(I, s_{max})$, *if $s$ is not locally optimal, and* $f_A(I,s) \leq t \leq T$, *if $s$ is locally optimal* $\}$
- $f_{\tilde{A}}(\langle s,t\rangle) = t$
- $opt_{\tilde{A}} = max$
- $N_{\tilde{A}}(I, \langle s,t\rangle) = \{\langle s', t+1\rangle$; *where $s$ is locally optimal,$s = s'$ and $t < T$, or $s$ is not locally optimal, $s = s'$ and $t < f_A(I, s_{max})$ or $s$ is not locally optimal,* $s' \in N_A(I,s)$ *and* $f_A(I,s') = t+1\}$

**Lemma 3.1** *Let $A \in PLS$, $\tilde{A}$ as defined above. Then $A \leq^{PLS} \tilde{A}$. For all $I \in I_A$ and for all $s \in \mathcal{FS}_A(I)$, let $s_{max} \in N_A(I,s)$ be a neighbor of $s$, such that $f_A(I, s_{max}) \geq max\{f_A(I,s')|s' \in N_A(I,s)\}$. Suppose that the value $f_A(I, s_{max})$ can be computed in polynomial time. Then $A \in PLS$.*

**Proof**
Since it can be determined in polynomial time for $I \in I_A$, whether $s \in \mathcal{FS}_A(I)$, and since $f_A(I, s_{max})$ can be computed in polynomial time, it can be determined in polynomial time whether $\langle s,t\rangle \in \mathcal{FS}_{\tilde{A}}(I)$. From this and the fact that $A \in$ PLS, it follows that $\tilde{A} \in$ PLS.
The PLS-reduction $\langle f, g\rangle$ from $A$ to $\tilde{A}$ is defined as

$$f(I) = I$$
$$g(I, \langle s, T\rangle) = s$$

Note that $f$ and $g$ can be computed in polynomial time. □

**Lemma 3.2** *There are $A \in PLS$, such that $\tilde{A}$ is PLS-complete.*

**Proof**
This corollary follows, since there exist PLS-complete problems $A$, for which $\forall I \in I_A$ and $\forall s \in \mathcal{FS}_A(I)$, $N_A(I,s)$ can be enumerated in polynomial time. The problem *CircuitFlip* is such a problem. Other problems are for instance *Satisfiability* with the Flip neighborhood, and *MaxCut* with the Swap neighborhood and *TSP* with the Lin-Kernighan neighborhood, see [Kre90], [SY91] and [Pap92]. □

**Theorem 3.3** *Let $A \in PLS$. Define $\tilde{A}$ as before. Then $\tilde{A}^* \in P$.*

**Proof**
The only locally optimal solutions in $\mathcal{FS}_{\tilde{A}}(I)$ are $\langle s, T\rangle$, where $s$ is locally optimal in $\mathcal{FS}_A(I)$. Given $\langle s, t\rangle$, there is a locally optimal solution reachable from $\langle s, t\rangle$ using a path with exactly $T - t$ vertices. Whether $d \geq (T - t)$ can be decided in polynomial time. □

**Acknowledgement** I want to thank Leen Torenvliet and Harry Buhrman for their questions and comments.

## References

[ALM+92] S. Arora, C Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. *33rd Annual Symposium on Foundations of Computer Science*, pages 14–23, 1992.

[BDG88] J.L. Balcázar, J. Diáz, and J. Gabarró. *Structural Complexity 1*. Springer-Verlag, 1988.

[JPY88] D.S. Johnson, C.H. Papadimitriou, and M. Yannakakis. How easy is local search? *J. Comput. System Sci.*, 37:79–100, 1988.

[Kre90] M.W. Krentel. On finding and verifying locally optimal solutions. *SIAM J. Comput.*, 19:742–749, 1990.

[Pap92] C.H. Papadimitriou. The complexity of the lin-kernighan algorithm. *SIAM J. Comput.*, pages 450–465, 1992.

[PSY90] C.H. Papadimitriou, A.A. Schäffer, and M. Yannakakis. On the complexity of local search. *Proc. 22nd Annual ACM Symposium on Theory of Computing, Association for Computer Machinery*, pages 438–445, 1990.

[SY91] A.A. Schäffer and M Yannakakis. Simple local search problems that are hard to solve. *SIAM J. Comput.*, 20:56–87, 1991.

# A The completeness of *CircuitFlip**

In this section, we give a reduction $g$ from $A$ to *CircuitFlip**, $\forall A \in$ NP. This proves that *CircuitFlip** is NP-complete.
Let $A \in$ NP, $M_A$ a non-deterministic polynomial time bounded Turing machine recognizing $A$. Assume that the running time of $M_A$ on input $x$ is bounded by $p_A(|x|)$. Note that $p_A(|x|)$ is also a bound on the length of a configuration of $M_A$ on $x$. Since the running time of $M_A$ is polynomially bounded, $p_A(|x|)$ is a polynomial in $|x|$. The reduction $g$ described here, computes from $M_A$, $p_A$ and $x$ a Boolean circuit $C_A$, an initial solution $s$ and a distance $d$. We start describing the Boolean circuit $C_A$. Let $T = 2^{2p_A(|x|)}$.
Consider the computation tree of $M_A$ on $x$. We assume that every configuration $c$ of $M_A$ is followed by exactly two configurations $c_1$ and $c_2$. We would like to translate computation paths in the computation tree of $M_A$ on $x$ to augmenting paths in $G_{CF}(C_A)$, where $G_{CF}(C_A)$ is the local search graph of $\mathcal{FS}(C_A)$. Furthermore, augmenting paths corresponding to computation paths that end in rejecting final configurations are distinctively longer than augmenting paths corresponding to computation paths that end in accepting final configurations.
Let $c, c_i$ be configurations of $M_A$. Either $i \in \{1,2\}$ and $M_A$ can go from $c$ to $c_i$ in one step or $c = c_i$ is a rejecting final configuration and $t < T$. The solutions $(c,t)$ and $(c_i, t+1)$ can differ in more than one bit. So they can not be neighbors in $\mathcal{FS}_{CF}(C_A)$. Instead, an augmenting path in $G_{CF}(C_A)$ exists from a solution associated with $(c,t)$ to a solution associated with $(c_i, t+1)$. The length of this computation path is polynomially bounded in $|x|$.
In general, a solution of $\mathcal{FS}_{CF}(C_A)$ is of the form $\langle (c,t),(c',t'),(\tilde{c},\tilde{t}), n_1, n_2, l_1, l_2\rangle$, where $c, c'$ and $\tilde{c}$ are sequences of $p_A(|x|)$ bits, $t, t'$ and $\tilde{t}$ are sequences of $\log T$ bits and $n_1, n_2, l_1$ and $l_2$ are bits. A solution associated with $(c,t)$ is of the form $s = \langle (c,t),(c,t),(c,t),0,0,0,0\rangle$. The three components $(c,t)$, $(c',t')$ and $(\tilde{c},\tilde{t})$ are needed to ensure that only augmenting paths exist from $s$ to $s'$, where $s$ is the solution in $\mathcal{FS}_{CF}(C_A)$ associated with $(c,t)$ and $s'$ is the solution in $\mathcal{FS}_{CF}(C_A)$ associated with $(c_i, t+1)$. The outputs of $C_A$ are of the form $\tau\gamma_1\gamma_2\gamma_3 b_0 b_1 b_2 b_3 b_4 \delta_1\delta_2\delta_3$, where $b_i$, $0 \le i \le 4$, are bits, $\tau$ is a sequence of $\log T$ bits and $\gamma_i, \delta_j$, $1 \le i,j \le 3$, are sequences of $p_A(|x|) + \log T$ bits.
The inputs of $C_A$ can be divided into a constant number of groups. Let $\alpha(|x|) = p_A(|x|) + \log T = 3p_A(|x|)$.

1. The first group consists of the *reasonable* inputs $s$. Input $s$ is of the form $s = \langle (c,t),(c,t),(c,t),0,0,0,0\rangle$, where $c$ is a configuration of $M_A$ of length $p_A(|x|)$ and $t \le T$. On input $s$, the output of $C_A$ is $\tau 0^{3\alpha(|x|)+5} 1^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

2. The second group consists of inputs $s$ of the form $s = \langle (c,t),(c,t),(c,t), n_1, n_2, 0, 0\rangle$, where $n_i = 1$, $n_j = 0$,($i \ne j$ and $i,j \in \{1,2\}$), $t < T$ and $c$ not an accepting final configuration. The output of $C_A$ on input $s$ is $\tau 0^{3\alpha(|x|)+4} 1 1^{3\alpha(|x|)+5}$, where $\tau$ is

the binary representation of $t$.

3. The third group consists of inputs $s$ of the form $s = \langle(c,t),(c,t),(\tilde{c},\tilde{t}),n_1,n_2,0,0\rangle$, where $n_i = 1, n_j = 0$, ($i \neq j$ and $i,j \in \{1,2\}$) and the hamming distance between $(\tilde{c},\tilde{t})$ and $(c_i, t+1)$ is $k$, $1 \leq k \leq \alpha(|x|)$. The output of $C_A$ on $s$ is $\tau 0^{2\alpha(|x|)+k} 1^{\alpha(|x|)-k} 000011^{3\alpha(|x|)+5}$.

4. The fourth group consists of inputs $s$ of the form $s = \langle(c,t),(c,t),(c_i,t+1),n_1,n_2,1,0\rangle$, where $n_i = 1, n_j = 0$, ($i \neq j$ and $i,j \in \{1,2\}$). The output of $C_A$ on $s$ is $\tau 0^{2\alpha(|x|)} 1^{\alpha(|x|)} 000111^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

5. The fifth group consists of inputs $s$ of the form $s = \langle(c,t),(c',t'),(c_i,t+1),n_1,n_2,1,0\rangle$, where $n_i = l_1 = 1, n_j = l_2 = 0$ and the hamming distance between $(c',t')$ and $(c_i,t+1)$ is k, $1 \leq k \leq \alpha(|x|)$. The output of $C_A$ on $s$ is $\tau 0^{\alpha(|x|)+k} 1^{2\alpha(|x|-k)} 000111^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

6. The sixth group consists of inputs $s$ of the form $s = \langle(c,t),(c_i,t+1),(c_i,t+1),n_1,n_2,1,1\rangle$, where $n_i = 1, n_j = 0$. The output of $C_A$ on $s$ is $\tau 0^{\alpha(|x|)} 1^{2\alpha(|x|)} 001111^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

7. The seventh group consists of inputs $s$ of the form $s = \langle(c',t'),(c_i,t+1),(c_i,t+1),n_1,n_2,1,1\rangle$, where $n_i = 1, n_j = 0$. Suppose that the hamming distance between $(c',t')$ and $(c_i,t+1)$ is $k$. The output of $C_A$ on $s$ is $\tau 0^k 1^{2\alpha(|x|)-k} 001111^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

8. The eighth group consists of inputs $s$ of the form $s = \langle(c_i,t+1),(c_i,t+1),(c_i,t+1),0,0,1,1\rangle$.
   The output of $C_A$ on $s$ is $\tau 1^{3\alpha(|x|)+4} 011111^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

9. The ninth group consists of inputs $s$ of the form $s = \langle(c_i,t+1),(c_i,t+1),(c_i,t+1),0,0,0,1\rangle$.
   The output of $C_A$ on $s$ is $\tau 1^{3\alpha(|x|)+5} 1^{3\alpha(|x|)+5}$, where $\tau$ is the binary representation of $t$.

10. The tenth group consists of all other possible inputs $s$ to $C_A$. To compute the output of $s$, let $\kappa$ be the hamming distance between $s$ and $\langle(c_0,1),(c_0,1),(c_0,1),0,0,0,0\rangle$. Then the output of $C_A$ on $s$ is $0^{\log T} 0^{3\alpha(|x|)+5} 1^{3\alpha(|x|)-\kappa} 0^{\kappa}$.

The following lemma determines the form of locally optimal solutions in $\mathcal{FS}_{CF}(C_A)$.

**Lemma A.1** *Every locally optimal solution in $\mathcal{FS}_{CF}(C_A)$ is of the form*

$$\langle(c,t),(c,t),(c,t),0,0,0,0\rangle,$$

*where $c$ is either an accepting final configuration, or $c$ is a rejecting final configuration and $t = T$.*

**Proof**
Let $s$ be an input of $C_A$, and therefore a solution in $\mathcal{FS}_{CF}(C_A)$. Suppose that $s$ does not belong to the first group. Then it is always possible to flip a bit of $s$ to improve the output of $C_A$. To see this consider every group, except the first group, separately. So in these cases $s$ can not be locally optimal.
Consider now $s = \langle (c,t),(c,t),(c,t),0,0,0,0\rangle$.
If $c$ is not a final configuration, flipping $n_1$ or $n_2$ improves the output.
If $c$ is a rejecting final configuration and $t < T$, flipping the bit $n_1$ improves the output of $C_A$ on $s$. □

The next lemma proves that computing steps of $M_A$ appear as polynomially bounded paths in $\mathcal{FS}_{CF}(C_A)$.

**Lemma A.2** *Let $c, c'$ be configurations of $M_A$ of length $p_A(|x|)$, and let $t$ be an integer, $1 \leq t < T$. Consider the solutions $s = \langle (c,t),(c,t),(c,t),0,0,0,0\rangle$ and $s' = \langle (c',t+1),(c',t+1),(c',t+1),0,0,0,0\rangle$ of $\mathcal{FS}_{CF}(C_A)$.*
*$M_A$ can go in one step from $c$ to $c'$ or $c$ is a rejecting final configuration if and only if there exists an augmenting path $p$ from $s$ to $s'$ such that intermediate vertices on $p$, which are inputs of $C_A$, do not belong to group 1 or group 10.*

**Proof**
Consider a solution $s = \langle (c,t),(c,t),(c,t),0,0,0,0\rangle$. Suppose that $c$ is not a final configuration. Let $x_1 = \langle (c',t+1),(c',t+1),(c',t+1),0,0,0,0\rangle$ and $x_2 = \langle (\tilde{c},t+1),(\tilde{c},t+1),(\tilde{c},t+1),0,0,0,0\rangle$, where $c'$ is the first configuration following $c$ and $\tilde{c}$ is the second configuration following $c$. Solution $s$ has exactly two neighbors $s_1, s_1'$ with a better cost than $s$. The solutions $s_1, s_1'$ are achieved by flipping respectively the bit $n_1, n_2$ from 0 to 1 in $s$.
We will show that all augmenting paths leaving $s_1$ pass $x_1$. In the same manner, it can be shown that all augmenting paths leaving $s_1'$ pass $x_2$.
Let $\tilde{s} = \langle (c_1,t_1),(c_2,t_2),(c_3,t_3),n_1,n_2,l_1,l_2\rangle$ be a solution not in group 10. If $n_1 = 1$, then flipping the $n_2$ bit results in a solution belonging to group 10. If $n_1 = 1$ and $l_1 = 0$, the first two components $(c_1,t_1),(c_2,t_2)$ must have the same value and $l_2 = 0$. So flipping a bit in $(c_1,t_1)$ or in $(c_2,t_2)$ results in a solution with a cost worse than $\tilde{s}$. Flipping the $n_1$ bit results in a solution belonging to either group 1 or group 10. In both cases, the resulting solution has a cost worse than $\tilde{s}$. Flipping bits in $(c_3,t_3)$ that increase the hamming distance between $(c_1',t_1+1)$ and $(c_3,t_3)$, where $c_1'$ is the first configuration following $c_1$, result in a solution with a cost worse than $s_1$. Finally, as long as $c_3$ is not the first configuration following $c_1$, flipping the $l_1$ bit results in a solution belonging to group 10.
Let solution $s_2 = \langle (c,t),(c,t),(c',t+1),1,0,0,0\rangle$, $\mathcal{HD}(s_1,s_2) = r$. Then every augmenting path leaving $s_1$ pass a sequence of vertices $u_1, u_2, \ldots, u_r = s_2$, in that order, where vertices $u_i$ satisfy the following properties.

1. $u_i = \langle (c,t),(c,t),(c_i,t_i),1,0,0,0\rangle$, where $1 \leq i < r$.

2. $\mathcal{HD}(u_{i-1}, u_i) = 1$, $1 \leq i \leq r$, where $u_0 = s_1$.

3. $\mathcal{HD}((c_{i-1}, t_{i-1}), (c', t+1)) > \mathcal{HD}((c_i, t_i), (c', t+1))$, $1 \leq i \leq r$, where $(c_0, t_0) = (c, t)$.

Vertex $s_2$ has exactly one neighbor $s_3$ with a better cost. The solution $s_3$ is achieved by flipping the $l_1$ bit from 0 to 1.
Let $\tilde{s} = \langle (c_1, t_1), (c_2, t_2), (c_3, t_3), n_1, n_2, l_1, l_2 \rangle$ be a solution not in group 10. If $n_1 = 1$, then flipping the $n_2$ bit from 0 to 1 results in a solution belonging to group 10. If $n_1 = l_1 = 1$ and $l_2 = 0$, then $c_3$ must be the first configuration following $c_1$, and $t_3 = t_1 + 1$. Flipping any bit in $c_3, t_3$ or $t_1$ destroys this relation. The resulting solutions belong to group 10. Flipping a bit in $c_1$ does not necessarily destroy this relation. But in that case, the cost of the resulting solution $\tilde{s}'$ is not better than the cost of $\tilde{s}$. So $\tilde{s}'$ is not on the same augmenting path as $\tilde{s}$. Flipping the $n_1$ bit results in a solution belonging to group 10. Flipping the $l_1$ bit results in a solution belonging to either group 4 or group 10. In both cases, the resulting solution has a cost worse than $\tilde{s}$. Flipping bits in $(c_2, t_2)$ that increase the hamming distance between $(c_2, t_2)$ and $(c_2, t_2)$ result in solutions with a cost worse than $\tilde{s}$. Finally, as long as $c_2$ has not the same value as $c_3$ or $t_2 \neq t_3$, $l_2$ has the value 0.
Let solution $s_4 = \langle (c, t), (c', t+1), (c', t+1), 1, 0, 0, 0 \rangle$. Suppose $\mathcal{HD}(s_3, s_4) = r'$. Then $s_3$ is followed on $p$ by a sequence of vertices $v_1, v_2, \ldots, v_{r'} = s_4$, in that order, where vertices $v_i$ satisfy the following properties.

1. $v_i = \langle (c, t), (c_i, t_i), (c', t+1), 1, 0, 1, 0 \rangle$, where $1 \leq i < r'$.

2. $\mathcal{HD}(v_{i-1}, v_i) = 1$, $1 \leq i \leq r'$, where $v_0 = s_3$.

3. $\mathcal{HD}((c_{i-1}, t_{i-1}), (c', t+1)) > \mathcal{HD}((c_i, t_i), (c', t+1))$, $1 \leq i \leq r'$, where $(c_0, t_0) = (c, t)$.

Solution $s_4$ has exactly one neighbor $s_5$ with a better solution. The solution $s_5$ is achieved by flipping the $l_2$ bit from 0 to 1.
Let $\tilde{s} = \langle (c_1, t_1), (c_2, t_2), (c_3, t_3), n_1, n_2, l_1, l_2 \rangle$ be a solution not in group 10. If $n_1 = 1$, then flipping the $n_2$ bit from 0 to 1 results in a solution belonging to group 10. If $n_1 = l_1 = l_2 = 1$, then the components $(c_2, t_2)$ and $(c_3, t_3)$ must have the same value. So flipping a bit in $(c_2, t_2)$ or $(c_3, t_3)$ results in a solution with a cost worse than $\tilde{s}$. Flipping the $l_1$ bit results in a solution belonging to group 10. Flipping the $l_2$ bit results in a solution belonging either to group 6 or group 10. In both cases the resulting solution has a cost worse than $\tilde{s}$. While $c_1 \neq c_2$ or $t_1 \neq t_2$, flipping the $n_1$ bit results in a solution belonging to group 10. Flipping bits in $(c_1, t_1)$ that increase the hamming distance between $(c_1, t_1)$ and $(c_2, t_2)$ results in a solution with a cost worse than $\tilde{s}$.
Consider solution $s_6 = \langle (c', t+1), (c', t+1), (c', t+1), 1, 0, 0, 0 \rangle$. Suppose $\mathcal{HD}(s_5, s_6) = \tilde{r}$. Then $s_5$ is followed on $p$ by a sequence of vertices $w_1, w_2, \ldots, w_{\tilde{r}} = s_6$, in that order, where vertices $w_i$ satisfy the following properties.

1. $w_i = \langle (c_i, t_i), (c', t+1), (c', t+1), 1, 0, 1, 1 \rangle$, where $1 \leq i < \tilde{r}$.

2. $\mathcal{HD}(w_{i-1}, w_i) = 1$, $1 \leq i \leq \tilde{r}$, where $w_0 = s_5$.

3. $\mathcal{HD}((c_{i-1}, t_{i-1}), (c', t+1)) > \mathcal{HD}((c_i, t_i), (c', t+1))$, $1 \leq i \leq \tilde{r}$, where $(c_0, t_0) = (c, t)$.

Solution $s_6$ has one neighbor $s_7$ with a better cost. The solution $s_7$ is achieved by flipping the $n_1$ bit from 1 to 0.
Solution $s_7$ has one neighbor $s_8$ with a better cost. The solution $s_8$ is achieved by flipping the $l_1$ bit from 1 to 0.
Solution $s_8$ has one neighbor $x_1$ with a better cost. The solution $x_1$ is achieved by flipping the $l_2$ bit from 1 to 0.
It is easy to see that no intermediate vertices on any augmenting path from $s_1$ to $x_1$ is a locally optimal solution. Since $s$ and $s_1$ can not be locally optimal, there is at least one augmenting path from $s$ to $x_1$.
Before, we assumed that $c$ was not a final configuration. Suppose that $c$ is a rejecting final configuration. Let $x_1 = \langle (c', t+1), (c', t+1), (c', t+1), 0, 0, 0, 0 \rangle$. In the same way as above it can be shown that there is an augmenting path leaving $s$ and pass $x_1$, and that all augmenting paths leaving $s$ pass $x_1$. □

**Theorem A.1** *CircuitFlip* is NP-complete.*

**Proof**
Let $A \in$ NP, $M_A$ a non-deterministic polynomial time bounded Turing machine recognizing $A$. Assume that the running time of $M_A$ on input $x$ is bounded by $p_A(|x|)$. Note that $p_A(|x|)$ is also a bound on the maximal length of a configuration of $M_A$ on $x$. Let $T = 2^{2p_A(|x|)}$ and $\alpha(|x|) = p_A(|x|) + \log T = 3p_A(|x|)$.
Define reduction $g$ from A to *CircuitFlip** as $f(x) = (C_A, s = \langle (c_0, 1), (c_0, 1), (c_0, 1), 0, 0, 0, 0 \rangle, p_A(|x|)(3\alpha(|x|) + 6))$, where $C_A$ is constructed as described before, $c_0$ is the initial configuration of $M_A$ on $x$ and $t$ is a sequence of $\log T$ bits. Note that for every input of $C_A$, the output of $C_A$ can be computed in deterministic polynomial time. Therefore, $C_A$ can be computed in polynomial time. For more details see [BDG88].
Suppose $x \in A$. Then there is a computation path $c_0, c_1, \ldots, c_t$ with $c_t$ an accepting final configuration and $t \leq p_A(|x|)$. Using lemma A.2, the solution $s' = \langle (c_t, t'), (c_t, t'), (c_t, t'), 0, 0, 0, 0 \rangle$ is reachable from $s$, and $t' \leq t(3\alpha(|x|) + 6) \leq p_A(|x|)(3\alpha(|x|) + 6)$. It is easy to see that $s'$ is local optimal. Therefore, $f(x) \in$ *CircuitFlip**.
Suppose $x \notin A$. Then every computation path in the computation tree of $M_A$ on $x$ reaches a rejecting final configuration. Using lemma A.2, every augmenting path $p$ in $G_{CF}(C_A)$ leaving $s$ reaches only locally optimal solutions of the form $\langle (c', T), (c', T), (c', T), 0, 0, 0, 0 \rangle$, with $c'$ a rejecting final configuration of $M_A$. The length of $p$ is $T > p_A(|x|)(3\alpha(|x|) + 6)$. So $f(x) \notin$ *CircuitFlip**. □

# The ILLC Prepublication Series

CT-91-10 John Tromp, Paul Vitányi — A Randomized Algorithm for Two-Process Wait-Free Test-and-Set
CT-91-11 Lane A. Hemachandra, Edith Spaan — Quasi-Injective Reductions
CT-91-12 Krzysztof R. Apt, Dino Pedreschi — Reasoning about Termination of Prolog Programs

*Computational Linguistics*

CL-91-01 J.C. Scholtes — Kohonen Feature Maps in Natural Language Processing
CL-91-02 J.C. Scholtes — Neural Nets and their Relevance for Information Retrieval
CL-91-03 Hub Prüst, Remko Scha, Martin van den Berg — A Formal Discourse Grammar tackling Verb Phrase Anaphora

*Other Prepublications*

X-91-01 Alexander Chagrov, Michael Zakharyaschev — The Disjunction Property of Intermediate Propositional Logics
X-91-02 Alexander Chagrov, Michael Zakharyaschev — On the Undecidability of the Disjunction Property of Intermediate Propositional Logics
X-91-03 V. Yu. Shavrukov — Subalgebras of Diagonalizable Algebras of Theories containing Arithmetic
X-91-04 K.N. Ignatiev — Partial Conservativity and Modal Logics
X-91-05 Johan van Benthem — Temporal Logic
X-91-06 — Annual Report 1990
X-91-07 A.S. Troelstra — Lectures on Linear Logic, Errata and Supplement
X-91-08 Giorgie Dzhaparidze — Logic of Tolerance
X-91-09 L.D. Beklemishev — On Bimodal Provability Logics for $\Pi_1$-axiomatized Extensions of Arithmetical Theories
X-91-10 Michiel van Lambalgen — Independence, Randomness and the Axiom of Choice
X-91-11 Michael Zakharyaschev — Canonical Formulas for K4. Part I: Basic Results
X-91-12 Herman Hendriks — Flexibele Categoriale Syntaxis en Semantiek: de proefschriften van Frans Zwarts en Michael Moortgat
X-91-13 Max I. Kanovich — The Multiplicative Fragment of Linear Logic is NP-Complete
X-91-14 Max I. Kanovich — The Horn Fragment of Linear Logic is NP-Complete
X-91-15 V. Yu. Shavrukov — Subalgebras of Diagonalizable Algebras of Theories containing Arithmetic, revised version
X-91-16 V.G. Kanovei — Undecidable Hypotheses in Edward Nelson's Internal Set Theory
X-91-17 Michiel van Lambalgen — Independence, Randomness and the Axiom of Choice, Revised Version
X-91-18 Giovanna Cepparello — New Semantics for Predicate Modal Logic: an Analysis from a standard point of view
X-91-19 Papers presented at the Provability Interpretability Arithmetic Conference, 24-31 Aug. 1991, Dept. of Phil., Utrecht University

## 1992

Annual Report 1991

*Logic, Semantics and Philosophy of Langauge*

LP-92-01 Víctor Sánchez Valencia — Lambek Grammar: an Information-based Categorial Grammar
LP-92-02 Patrick Blackburn — Modal Logic and Attribute Value Structures
LP-92-03 Szabolcs Mikulás — The Completeness of the Lambek Calculus with respect to Relational Semantics
LP-92-04 Paul Dekker — An Update Semantics for Dynamic Predicate Logic
LP-92-05 David I. Beaver — The Kinematics of Presupposition
LP-92-06 Patrick Blackburn, Edith Spaan — A Modal Perspective on the Computational Complexity of Attribute Value Grammar
LP-92-07 Jeroen Groenendijk, Martin Stokhof — A Note on Interrogatives and Adverbs of Quantification
LP-92-08 Maarten de Rijke — A System of Dynamic Modal Logic
LP-92-09 Johan van Benthem — Quantifiers in the world of Types
LP-92-10 Maarten de Rijke — Meeting Some Neighbours (a dynamic modal logic meets theories of change and knowledge representation)
LP-92-11 Johan van Benthem — A note on Dynamic Arrow Logic
LP-92-12 Heinrich Wansing — Sequent Caluli for Normal Modal Propositional Logics
LP-92-13 Dag Westerståhl — Iterated Quantifiers
LP-92-14 Jeroen Groenendijk, Martin Stokhof — Interrogatives and Adverbs of Quantification

*Mathematical Logic and Foundations*

ML-92-01 A.S. Troelstra — Comparing the theory of Representations and Constructive Mathematics
ML-92-02 Dmitrij P. Skvortsov, Valentin B. Shehtman — Maximal Kripke-type Semantics for Modal and Superintuitionistic Predicate Logics
ML-92-03 Zoran Marković — On the Structure of Kripke Models of Heyting Arithmetic
ML-92-04 Dimiter Vakarelov — A Modal Theory of Arrows, Arrow Logics I
ML-92-05 Domenico Zambella — Shavrukov's Theorem on the Subalgebras of Diagonalizable Algebras for Theories containing $I\Delta_0$ + EXP
ML-92-06 D.M. Gabbay, Valentin B. Shehtman — Undecidability of Modal and Intermediate First-Order Logics with Two Individual Variables
ML-92-07 Harold Schellinx — How to Broaden your Horizon
ML-92-08 Raymond Hoofman — Information Systems as Coalgebras
ML-92-09 A.S. Troelstra — Realizability
ML-92-10 V.Yu. Shavrukov — A Smart Child of Peano's

*Compution and Complexity Theory*

CT-92-01 Erik de Haas, Peter van Emde Boas — Object Oriented Application Flow Graphs and their Semantics
CT-92-02 Karen L. Kwast, Sieger van Denneheuvel — Weak Equivalence: Theory and Applications
CT-92-03 Krzysztof R. Apt, Kees Doets — A new Definition of SLDNF-resolution

*Other Prepublications*

X-92-01 Heinrich Wansing — The Logic of Information Structures
X-92-02 Konstantin N. Ignatiev — The Closed Fragment of Dzhaparidze's Polymodal Logic and the Logic of $\Sigma_1$ conservativity
X-92-03 Willem Groeneveld — Dynamic Semantics and Circular Propositions, revised version
X-92-04 Johan van Benthem — Modeling the Kinematics of Meaning
X-92-05 Erik de Haas, Peter van Emde Boas — Object Oriented Application Flow Graphs and their Semantics, revised version

## 1993

*Logic, Semantics and Philosophy of Langauge*

LP-93-01 Martijn Spaan — Parallel Quantification
LP-93-02 Makoto Kanazawa — Dynamic Generalized Quantifiers and Monotonicity

*Mathematical Logic and Foundations*

ML-93-01 Maciej Kandulski — Commutative Lambek Categorial Grammars
ML-93-02 Johan van Benthem, Natasha Alechina — Modal Quantification over Structured Domains
ML-93-03 Mati Pentus — The Conjoinablity Relation in Lambek Calculus and Linear Logic
ML-93-04 Andreja Prijatelj — Bounded Contraction and Many-Valued Semantics
ML-93-05 Raymond Hoofman, Harold Schellinx — Models of the Untyped $\lambda$-calculus in Semi Cartesian Closed Categories
ML-93-06 J. Zashev — Categorial Generalization of Algebraic Recursion Theory

*Compution and Complexity Theory*

CT-93-01 Marianne Kalsbeek — The Vanilla Meta-Interpreter for Definite Logic Programs and Ambivalent Syntax
CT-93-02 Sophie Fischer — A Note on the Complexity of Local Search problems

*Other Prepublications*

X-93-01 Paul Dekker — Existential Disclosure, revised version