K1,3-free and W4-free graphs

Eindhoven University of Technology

Department of Mathematics and Computing Science

$K_{1,3}$-free and W_4-free graphs

by

T. Kloks

94/25

Computing Science Note 94/25
Eindhoven, May 1994

COMPUTING SCIENCE NOTES

This is a series of notes of the Computing Science Section of the Department of Mathematics and Computing Science Eindhoven University of Technology.
Since many of these notes are preliminary versions or may be published elsewhere, they have a limited distribution only and are not for review.
Copies of these notes are available from the author.

Copies can be ordered from:
Mrs. M. Philips
Eindhoven University of Technology
Department of Mathematics and Computing Science
P.O. Box 513
5600 MB EINDHOVEN
The Netherlands
ISSN 0926-4515

$K_{1,3}$-free and W_4-free graphs

T. Kloks *

Department of Mathematics and Computing Science

Eindhoven University of Technology

P.O.Box 513, 5600 MB Eindhoven, The Netherlands

Abstract

We show that in any claw- and 4-wheel-free graph the number of maximal cliques is polynomial bounded. We show there exists an efficient algorithm to solve the MAXIMUM CLIQUE problem for this class of graphs. We also give an $O(ne)$ recognition algorithm.

1 Introduction

A claw and a 4-wheel are depicted in Figure 1.

Figure 1: The 'claw' $K_{1,3}$ (left) and the '4-wheel' W_4 (right)

We consider graphs without claw $K_{1,3}$ or 4-wheel W_4 as an induced subgraph. This research was motivated by recent results on a graph class which we called *dominoes* ([12]). A domino is a graph in which every vertex is contained in at most two maximal cliques. Dominoes can be characterized as those graphs without induced claw, gem or 4-wheel. They are exactly the line graphs of multigraphs without triangles (see also [2, 1]).

Clearly, a domino can have at most n maximal cliques, where n is the number of vertices in the graph. One possible generalization of the dominoes

*Email: ton@win.tue.nl

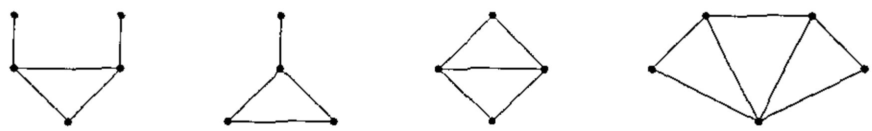

Figure 2: 'Bull', 'paw', 'diamond' and 'gem'

is the following. Consider the class of graphs without claw or gem (Figure 2). Hence the neighborhood of each vertex is P_4-free (P_4 is the path with four vertices), i.e., each neighborhood induces a *cograph* (see [5]). In fact, since the graph is also claw-free, each neighborhood must have independence number $\alpha \leq 2$. An example of a cograph with $2t$ vertices and with $\alpha \leq 2$ is a cocktail party graph, i.e., the complement of tK_2. It is easily seen that this graph has 2^t different maximal cliques. Hence this generalization of dominoes does not preserve the polynomial bound on the number of maximal cliques. In the next section we give a complete characterization of claw-free and gem-free graphs.

In this paper we look at another generalization of the dominoes, i.e., the class of graphs without induced claw or 4-wheel. We show that a graph in this class can have only a polynomial number of maximal cliques. We show that the MAXIMUM CLIQUE problem can be solved efficiently and we give an efficient recognition algorithm.

2 Related results

In this section we discuss some related results. We give characterizations of graphs without claws or triangles, without claws or paws, without claws or diamonds and without claws or gems. For the different forbidden induced subgraphs mentioned in this paper we refer to Figure 2.

Definition 1 *A graph* H *is obtained from a graph* G *by* duplication *if it can be obtained by substituting cliques for some of the vertices of* G.

We mention here a related result of Shepherd's [19] of claw-free and bull-free graphs.

Lemma 1 *A connected graph* G *is claw-free and bull-free if and only if either* $\alpha(G) = 2$ *or* G *is obtained from a path or cycle by duplication.*

2.1 Claw-free and triangle-free

We start with graphs without claws or triangles. Clearly, in such a graph every vertex can have degree at most two. Hence we obtain the following result.

Lemma 2 *A graph* G *is claw-free and triangle-free if and only if every connected component of* G *is a path or a cycle of length at least four.*

2.2 Claw-free and paw-free

We can use the following characterization of Olariu's ([14]).

Lemma 3 *A graph* G *is paw-free if and only if each connected component of* G *is either triangle-free or complete multipartite.*

If a connected complete multipartite graph has a color class with more than two vertices, it contains a claw. Hence every color class in the complete multipartite graph can have at most two vertices. Using Lemma 2 we obtain the following characterization.

Lemma 4 *A graph* G *is claw-free and paw-free if and only if every component is either a path, a cycle, or a complete multipartite graph in which every color class has at most two vertices.*

It follows that recognizing this class of graphs can be done in linear time (i.e., in $O(n + e)$). Notice that a graph is complete multipartite with $\alpha \leq 2$ if and only if every vertex is non adjacent to at most one other vertex, i.e., the graph is a cocktail party graph with some additional vertices made adjacent to all other vertices.

2.3 Claw-free and diamond-free

The reverse operation of duplication is taking the representative. For a vertex x let $N(x)$ be the set of neighbors, and let $N[x]$ be the *closed* neighborhood of x, i.e., $N[x] = \{x\} \cup N(x)$.

Definition 2 *The* representative *of a graph* G *is the graph* H *obtained by identifying vertices with the same closed neighborhood. Vertices with the same closed neighborhood are called equivalent.*

As mentioned earlier, a domino is the line graph of a multigraph without triangles. In [12] the following result was obtained.

Lemma 5 *A graph* H *is a representative of a domino if and only if* H *is claw-free and diamond-free and no two vertices of* H *have the same closed neighborhood.*

Hence, the representative of a claw-free and diamond-free graph is the line graph of triangle-free graph.

A vertex is *simplicial* if its neighborhood is a clique. Notice, that duplication of a vertex that is *not* simplicial results in a diamond. Hence we obtain the following result.

Lemma 6 *A graph* G *is claw-free and diamond-free if and only if* G *is obtained from the line graph of a triangle-free graph by duplication of simplicial vertices.*

Computing the representative H of a graph G can be performed in linear time [12]. Next checking whether H is the line graph of a triangle-free graph also can be done in linear time [12]. Finally, checking whether or not only simplicial vertices of H are duplicated can be done in linear time using the fact that one can compute for each vertex of H the maximal cliques it is contained in in linear time [12]. Hence, recognizing claw-free and diamond-free graphs can be done in linear time.

2.4 Claw-free and gem-free

A cograph is a P_4-free graph, i.e., a graph without an induced path with four vertices. A characterization of cographs found by Seinsche [18] is the following.

Lemma 7 *A graph is a cograph if and only if every nontrivial induced subgraph or its complement is disconnected.*

Lemma 8 *A graph* G *is claw-free and gem-free if and only if for every vertex the neighborhood is the complement of a disjoint union of complete bipartite graphs and isolated vertices.*

Proof. Clearly, if each neighborhood is the complement of a disjoint union of complete bipartite graphs and isolated vertices, the graph is claw-free and gem-free.

Now assume the graph is claw-free and gem-free and consider a neighborhood $N(x)$. Then this neighborhood is a cograph with independence number $\alpha \leq 2$. If the neighborhood is disconnected, it must be the disjoint union of two cliques, i.e., the complement of a complete bipartite graph. Henceforth, assume the neighborhood is connected. Then by Lemma 7, the complement is disconnected. Each component C of the complement must be triangle-free. Now either C consists of one vertex, or the complement is disconnected. But the complement of C must satisfy $\alpha \leq 2$, and hence if C is not one vertex, the complement of C must be must be the disjoint union of two cliques, i.e., C is a complete bipartite graph. This proves the lemma. □

Using the characterization of Lemma 8 the class of claw-free and gem-free graphs can easily be recognized in $O(ne)$ time.

3 A characterization of $K_{1,3}$-free and W_4-free graphs

In this section we characterize claw-free and 4-wheel-free graphs by their neighborhoods.

Definition 3 *A bipartite graph* $G(A \cup B, E)$ *is a* chain graph *if for every pair of vertices* $x, y \in A$ *either* $N(x) \subseteq N(y)$ *or* $N(y) \subseteq N(x)$.

Notice that the choice of the color class A in this definition is not essential.

Chain graphs were introduced by Yannakakis [21]. For a bipartite graph $G(A \cup B, E)$ let $C(G)$ be the graph obtained by adding edges to make cliques of the two color classes. A graph is called *chordal* if it has no induced chordless cycle of length larger than three. The following result was obtained in [21].

Lemma 9 *A bipartite graph* G *is a chain graph if and only if* $C(G)$ *is chordal.*

Notice that this is equivalent with: A bipartite graph is a chain graph if and only if its complement is chordal.

Theorem 1 *A graph is claw-free and 4-wheel-free if and only if each neighborhood is either obtained from* C_5 *or from* W_5 *by duplication, or it is the complement of a chain graph.*

Proof. Notice that if each neighborhood is either obtained from C_5 by duplication or it is the complement of a chain graph, then the graph is claw-free and 4-wheel-free.

Assume G is claw-free and 4-wheel-free and consider a vertex x.

First assume $N(x)$ has a chordless cycle C with at least four vertices. Since the graph is claw-free, $\alpha(N(x)) \leq 2$ and hence this cycle can have length at most five, Since the graph is 4-wheel-free, $N(x)$ cannot have a 4-cycle. Hence C is a 5-cycle. Let $a_1, \ldots, a_5$ be the vertices of C, with a_i and a_{i+1} adjacent for $i = 1, \ldots, 5$ taken modulo 5.

Let $y \in N(x) \setminus C$. Since $\alpha(N(x)) \leq 2$ y is adjacent to at least one vertex of every non adjacent pair of vertices of C. Notice that y cannot be adjacent to four vertices of C, since otherwise $N(x)$ has a 4-cycle. It follows that y is either adjacent to all vertices of C or to a_{i-1}, a_i and a_{i+1} for some i. Let A_i be the set of vertices of $N(x) \setminus C$ which are adjacent exactly to a_{i-1}, a_i and a_{i+1} and let A be the set of vertices of $N(x) \setminus C$ which are adjacent to all vertices of C.

Claim *The sets* A *and* A_i *(*$i = 1, \ldots, 5$*) are cliques.*

Proof of claim. Every pair of vertices of A_i is adjacent to non adjacent vertices a_{i-1} and a_{i+1}. Hence every pair of vertices of A_i is adjacent, since otherwise there would be a 4-cycle. The same holds for the set A. $\square$

Claim *Every vertex of* A_i *is adjacent to every vertex of* A_{i+1}.

Proof of claim. Consider a vertex $p \in A_i$ and a vertex $q \in A_{i+1}$. Then p and q must be adjacent since otherwise a_{i-2} p and q are an independent set of size three. $\square$

Claim *Every vertex of* A_{i-1} *is non adjacent to every vertex of* A_{i+1}.

Proof of claim. Let $p \in A_{i-1}$ and $q \in A_{i+1}$. Then p and q cannot be adjacent, since otherwise we would obtain a chordless 4-cycle $\{p, q, a_{i+2}, a_{i+3}\}$. $\square$

Claim *Every vertex of* A *is adjacent to every vertex of* A_i.

Proof of claim. Consider $p \in A$ and $q \in A_i$. Then p and q must be adjacent, otherwise $\{p, a_{i-1}, q, a_{i+1}\}$ would be a chordless 4-cycle. $\square$

These observations show that in case $N(x)$ contains a cycle it is obtained from C_5 (if $A = \emptyset$) or from W_5 (if $A \neq \emptyset$) by duplication.

Now assume that $N(x)$ does not have a chordless cycle of length at least four, i.e., $N(x)$ is chordal. Consider the complement of $N(x)$. The complement can not contain a chordless cycle of length at least six, since otherwise, $N(x)$ has a chordless 4-cycle. Clearly, the complement of $N(x)$ cannot contain a C_5 otherwise $N(x)$ also contains this. Since $\alpha(N(x)) \leq 2$, the complement of $N(x)$ does not contain a triangle. It follows that the complement of $N(x)$ is bipartite. By Lemma 9 this proves the theorem. $\square$

Remark 1 *Another way to prove Theorem 1 is to use a result of Fouquet [7]. If* G *is a claw-free graph with* $\alpha(G) \geq 3$ *then every neighborhood either contains an induced* C_5 *or is covered by two complete graphs.*

An alternative way to characterize claw-free and 4-wheel-free graphs is the following.

Lemma 10 *A graph* G *is claw-free and 4-wheel-free if and only if in the representative* H *of* G *every neighborhood is either* C_5 *or the complement of a chain graph.*

Proof. It is easy to see that G is claw-free and 4-wheel-free if and only if this holds for its representative H.

Assume some vertex x of G has a 5-wheel in its neighborhood, and let $y \in N(x)$ be such that $N[x] \subseteq N[y]$. We claim that x and y are equivalent. This follows immediately from Theorem 1 since the neighborhood of y contains a W_5 with x as a central vertex. Hence $N[y] \subseteq N[x]$. This shows that in the representative each neighborhood is either obtained from C_5 by duplication or is the complement of a chain graph.

Assume x has a C_5 in its neighborhood in G and let $A_1, \ldots, A_5$ be the equivalence classes of this C_5 (with A_i adjacent to A_{i+1}). Assume A_1 has at least two vertices p and q and assume p has some neighbor $z \in N(p) \setminus N(q)$. Then z must be adjacent to all vertices of A_2 or to all vertices of A_5 otherwise we have a claw. Assume z is adjacent to all vertices of A_2. Now any vertex of A_2 together q, z and a vertex of A_3 induces a claw (z cannot be adjacent to a vertex of A_3 otherwise x and z have nonadjacent common neighbors). Hence all vertices of A_1 are equivalent. This proves the theorem. □

4 Recognition of $K_{1,3}$-free and W_4-free graphs

We can use Theorem 1 to obtain an $O(ne)$ recognition algorithm. For each vertex x compute the representative of the graph induced by the neighborhood and check if it is C_5, W_5 or the complement of a chain graph. Computing for each vertex x the subgraph of G induced by $N(x)$, can easily be done in $O(nd(x))$ time (construct new adjacency lists for each vertex of $N(x)$). For each neighborhood $N(x)$ we can compute the representative in linear time [12], i.e., in $O(nd(x))$. Checking if this graph is W_5 or C_5 clearly takes constant time.

Finally, there are several ways to recognize complements of chain graphs. One way of doing this is to check if the graph is chordal (which takes linear time [17, 20]) and then check if the complement is bipartite. Hence this can be done in $O(nd(x))$ time. Alternatively, instead of testing chordality we could check whether the graph is an interval graph, which also takes linear time [4].

In total, summing over all vertices of the graph, we get an $O(ne)$ recognition algorithm for $K_{1,3}$-free and W_4-free graphs.

Lemma 11 *There exists an $O(ne)$ recognition algorithm for $K_{1,3}$-free and W_4-free graphs with n vertices and e edges.*

At the moment we do not know whether this result is best possible. We mention that, to our knowledge, the best time bound to test if a graph is claw-free is $O(en^{\alpha-1})$ (using $O(n^2)$ space). (Here $O(n^\alpha)$ is the time needed to do a matrix multiplication). This is done by doing a fast matrix multiplication for the complement of each neighborhood to test whether this contains a triangle.

5 maximum clique for claw-free and 4-wheel-free graphs

A chordal graph with n vertices can have at most n maximal cliques (with equality only if the graph has no edges). This was first pointed out in [8]. Hence, by Theorem 1 we obtain the following result.

Corollary 1 *If* G *is claw-free and 4-wheel-free then every vertex* x *is contained in at most* $d(x)$ *maximal cliques with equality only if* $N(x)$ *is* C_5*,* W_5 *or two isolated vertices. (Here* $d(x)$ *is the degree of* x*.) Hence, if* G *is non trivial and connected it has at most* $2e$ *maximal cliques.*

Now it is easy to compute the maximum clique in a claw-free and 4-wheel-free graph. This is of interest since the MAXIMUM CLIQUE problem is NP-complete for claw-free graphs in general. This follows from the fact that MAXIMUM INDEPENDENT SET problem is NP-complete for triangle-free graphs [16].

We compute for each vertex x the maximum clique size contained in the neighborhood $N(x)$ of x, as follows. First we can determine the representative of the subgraph induced by $N(x)$ in linear time. Each vertex in the representative has a weight attached to it, which is equal to the number of vertices in its equivalence class [12]. We solve the WEIGHTED MAXIMUM CLIQUE problem for the representative. If the representative is C_5 or W_5 this obviously takes constant time.

If the representative is the complement of a chain graph, we can solve the MAXIMUM WEIGHTED CLIQUE problem by using the algorithm for chordal graphs of [9] (listing all maximal cliques), which takes linear time. This shows the following.

Lemma 12 *There exists an* $O(ne)$ *algorithm which computes a maximum clique in a* $K_{1,3}$*-free and* W_4*-free graph with* n *vertices and* e *edges.*

6 Conclusions

In this paper we considered the class of graphs which are both claw-free and 4-wheel-free. Our main result is a characterization of this class by the neighborhoods of the vertices. This characterization allows an efficient recognition algorithm. The characterization also shows that the MAXIMUM

CLIQUE problem can be solved efficiently for this class of graphs. In fact, it follows that the number of maximal cliques is linear for this class of graphs. This is of interest since the problem is NP-complete when restricted to the class of claw-free graphs.

We do not know whether similar results can be obtained for other classes of claw-free graphs. Exceptions are of course the claw-free graphs without odd hole or antihole for which, for example the MAXIMUM CLIQUE problem, can be solved in polynomial time, since claw-free graphs satisfy the strong perfect graph conjecture [15, 3]. In [7] Fouquet shows that claw-free and W_5-free graphs with independence number at least three can be recognized in $O(n^{\alpha} + ne)$ time. To us, it is not clear whether the MAXIMUM CLIQUE problem can be solved in polynomial time for this class of graphs.

7 Acknowledgements

We thank Annelies Jacobs for doing a splendid bibliographical research and Dieter Kratsch for his careful reading of the manuscript.

References

[1] Beineke, L. W. and R. J. Wilson, *Selected topics in graph theory*, Academic Press, 1978.

[2] Berge, C., *Hypergraphs*, North Holland, 1989.

[3] Berge, C. and C. Chvatal, *Topics on Perfect Graphs*, Ann. Disc. Math. **21**, 1984.

[4] Booth, K. S. and G. S. Lueker, Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms, *Journal of Computer and System Sciences* **13**, ('976), pp. 335–379.

[5] Corneil, D. G., Y. Perl and L. K. Stewart, A linear time recognition algorithm for cographs, *SIAM J. Comput.* **14**, (1985), pp. 926–934.

[6] Chiba, N. and T. Nishizeki, Arboricity and subgraph listing algorithms, *SIAM J. Comput.* **14**, (1985), pp. 210–223.

[7] Fouquet, J. L., A strengthening of Ben Rebea's lemma, *Journal of Combinatorial Theory, Series B* **59**, (1993), pp. 35–40.

[8] Fulkerson, D. R. and O. A. Gross, Incidence matrices and interval graphs, *Pacific J. Math.* **15**, (1965), pp. 835–855.

[9] Golumbic, M. C., *Algorithmic graph theory and perfect graphs*, Academic Press, New York, 1980.

[10] Itai, A. and M. Rodeh, Finding a minimal circuit in a graph, *SIAM J. Comput.* **7**, (1978), pp. 413–423.

[11] Kloks, T., *Treewidth*, Ph.D. Thesis, Utrecht University, Utrecht, The Netherlands, 1993.

[12] Kloks, T., D. Kratsch and H. Müller, Dominoes, *Computing Science Notes* 94/12, Eindhoven University of Technology, Eindhoven, The Netherlands, (1994).

[13] Leeuwen, J. van, Graph Algorithms. In: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, A: Algorithms and Complexity*, Elsevier Science Publ., Amsterdam, 1990, pp. 527–631.

[14] Olariu, S., Paw-free graphs, *Information Processing Letters* **28**, (1988), pp. 53–54.

[15] Parthasarathy, K. R. and G. Ravindra, The strong perfect graph conjecture is true for $K_{1,3}$-free graphs, *Journal of Combinatorial Theory, Series B* **21**, (9176), pp. 212–223.

[16] Poljak, S., A note on stable sets and colorings of graphs, *Comment. Math. Univ. Carolin.* **15**, (1974), pp. 307–309.

[17] Rose, D. J., R. E. Tarjan and G. S. Lueker, Algorithmic aspects of vertex elimination on graphs, *SIAM J. Comput.* **5**, (1976), pp. 266–283.

[18] Seinsche, D., On a property of the class of n-colorable graphs, *Journal of Combinatorial Theory, Series B* **16**, (1974), pp. 191–193.

[19] Shepherd, F. B., Hamiltonicity in claw-free graphs, *Journal of Combinatorial Theory, Series B* **53**, (1991), pp. 173–194.

[20] Tarjan, R. E. and M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, *SIAM J. Comput.* **13**, (1984), pp. 566–579.

[21] Yannakakis, M., Computing the minimum fill-in is NP-complete, *SIAM J. Alg. Disc. Meth.* **2**, (1981), pp. 77–79.

Computing Science Notes | **Department of Mathematics and Computing Science**
Eindhoven University of Technology

In this series appeared:

91/01	D. Alstein	Dynamic Reconfiguration in Distributed Hard Real-Time Systems, p. 14.
91/02	R.P. Nederpelt H.C.M. de Swart	Implication. A survey of the different logical analyses "if...,then...", p. 26.
91/03	J.P. Katoen L.A.M. Schoenmakers	Parallel Programs for the Recognition of *P*-invariant Segments, p. 16.
91/04	E. v.d. Sluis A.F. v.d. Stappen	Performance Analysis of VLSI Programs, p. 31.
91/05	D. de Reus	An Implementation Model for GOOD, p. 18.
91/06	K.M. van Hee	SPECIFICATIEMETHODEN, een overzicht, p. 20.
91/07	E.Poll	CPO-models for second order lambda calculus with recursive types and subtyping, p. 49.
91/08	H. Schepers	Terminology and Paradigms for Fault Tolerance, p. 25.
91/09	W.M.P.v.d.Aalst	Interval Timed Petri Nets and their analysis, p.53.
91/10	R.C.Backhouse P.J. de Bruin P. Hoogendijk G. Malcolm E. Voermans J. v.d. Woude	POLYNOMIAL RELATORS, p. 52.
91/11	R.C. Backhouse P.J. de Bruin G.Malcolm E.Voermans J. van der Woude	Relational Catamorphism, p. 31.
91/12	E. van der Sluis	A parallel local search algorithm for the travelling salesman problem, p. 12.
91/13	F. Rietman	A note on Extensionality, p. 21.
91/14	P. Lemmens	The PDB Hypermedia Package. Why and how it was built, p. 63.
91/15	A.T.M. Aerts K.M. van Hee	Eldorado: Architecture of a Functional Database Management System, p. 19.
91/16	A.J.J.M. Marcelis	An example of proving attribute grammars correct: the representation of arithmetical expressions by DAGs, p. 25.

91/17	A.T.M. Aerts P.M.E. de Bra K.M. van Hee	Transforming Functional Database Schemes to Relational Representations, p. 21.
91/18	Rik van Geldrop	Transformational Query Solving, p. 35.
91/19	Erik Poll	Some categorical properties for a model for second order lambda calculus with subtyping, p. 21.
91/20	A.E. Eiben R.V. Schuwer	Knowledge Base Systems, a Formal Model, p. 21.
91/21	J. Coenen W.-P. de Roever J.Zwiers	Assertional Data Reification Proofs: Survey and Perspective, p. 18.
91/22	G. Wolf	Schedule Management: an Object Oriented Approach, p. 26.
91/23	K.M. van Hee L.J. Somers M. Voorhoeve	Z and high level Petri nets, p. 16.
91/24	A.T.M. Aerts D. de Reus	Formal semantics for BRM with examples, p. 25.
91/25	P. Zhou J. Hooman R. Kuiper	A compositional proof system for real-time systems based on explicit clock temporal logic: soundness and complete ness, p. 52.
91/26	P. de Bra G.J. Houben J. Paredaens	The GOOD based hypertext reference model, p. 12.
91/27	F. de Boer C. Palamidessi	Embedding as a tool for language comparison: On the CSP hierarchy, p. 17.
91/28	F. de Boer	A compositional proof system for dynamic proces creation, p. 24.
91/29	H. Ten Eikelder R. van Geldrop	Correctness of Acceptor Schemes for Regular Languages, p. 31.
91/30	J.C.M. Baeten F.W. Vaandrager	An Algebra for Process Creation, p. 29.
91/31	H. ten Eikelder	Some algorithms to decide the equivalence of recursive types, p. 26.
91/32	P. Struik	Techniques for designing efficient parallel programs, p. 14.
91/33	W. v.d. Aalst	The modelling and analysis of queueing systems with QNM-ExSpect, p. 23.
91/34	J. Coenen	Specifying fault tolerant programs in deontic logic, p. 15.

91/35	F.S. de Boer J.W. Klop C. Palamidessi	Asynchronous communication in process algebra, p. 20.
92/01	J. Coenen J. Zwiers W.-P. de Roever	A note on compositional refinement, p. 27.
92/02	J. Coenen J. Hooman	A compositional semantics for fault tolerant real-time systems, p. 18.
92/03	J.C.M. Baeten J.A. Bergstra	Real space process algebra, p. 42.
92/04	J.P.H.W.v.d.Eijnde	Program derivation in acyclic graphs and related problems, p. 90.
92/05	J.P.H.W.v.d.Eijnde	Conservative fixpoint functions on a graph, p. 25.
92/06	J.C.M. Baeten J.A. Bergstra	Discrete time process algebra, p.45.
92/07	R.P. Nederpelt	The fine-structure of lambda calculus, p. 110.
92/08	R.P. Nederpelt F. Kamareddine	On stepwise explicit substitution, p. 30.
92/09	R.C. Backhouse	Calculating the Warshall/Floyd path algorithm, p. 14.
92/10	P.M.P. Rambags	Composition and decomposition in a CPN model, p. 55.
92/11	R.C. Backhouse J.S.C.P.v.d.Woude	Demonic operators and monotype factors, p. 29.
92/12	F. Kamareddine	Set theory and nominalisation, Part I, p.26.
92/13	F. Kamareddine	Set theory and nominalisation, Part II, p.22.
92/14	J.C.M. Baeten	The total order assumption, p. 10.
92/15	F. Kamareddine	A system at the cross-roads of functional and logic programming, p.36.
92/16	R.R. Seljée	Integrity checking in deductive databases; an exposition, p.32.
92/17	W.M.P. van der Aalst	Interval timed coloured Petri nets and their analysis, p. 20.
92/18	R.Nederpelt F. Kamareddine	A unified approach to Type Theory through a refined lambda-calculus, p. 30.
92/19	J.C.M.Baeten J.A.Bergstra S.A.Smolka	Axiomatizing Probabilistic Processes: ACP with Generative Probabilities, p. 36.
92/20	F.Kamareddine	Are Types for Natural Language? P. 32.

92/21	F.Kamareddine	Non well-foundedness and type freeness can unify the interpretation of functional application, p. 16.
92/22	R. Nederpelt F.Kamareddine	A useful lambda notation, p. 17.
92/23	F.Kamareddine E.Klein	Nominalization, Predication and Type Containment, p. 40.
92/24	M.Codish D.Dams Eyal Yardeni	Bottum-up Abstract Interpretation of Logic Programs, p. 33.
92/25	E.Poll	A Programming Logic for $F\omega$, p. 15.
92/26	T.H.W.Beelen W.J.J.Stut P.A.C.Verkoulen	A modelling method using MOVIE and SimCon/ExSpect, p. 15.
92/27	B. Watson G. Zwaan	A taxonomy of keyword pattern matching algorithms, p. 50.
93/01	R. van Geldrop	Deriving the Aho-Corasick algorithms: a case study into the synergy of programming methods, p. 36.
93/02	T. Verhoeff	A continuous version of the Prisoner's Dilemma, p. 17
93/03	T. Verhoeff	Quicksort for linked lists, p. 8.
93/04	E.H.L. Aarts J.H.M. Korst P.J. Zwietering	Deterministic and randomized local search, p. 78.
93/05	J.C.M. Baeten C. Verhoef	A congruence theorem for structured operational semantics with predicates, p. 18.
93/06	J.P. Veltkamp	On the unavoidability of metastable behaviour, p. 29
93/07	P.D. Moerland	Exercises in Multiprogramming, p. 97
93/08	J. Verhoosel	A Formal Deterministic Scheduling Model for Hard Real-Time Executions in DEDOS, p. 32.
93/09	K.M. van Hee	Systems Engineering: a Formal Approach Part I: System Concepts, p. 72.
93/10	K.M. van Hee	Systems Engineering: a Formal Approach Part II: Frameworks, p. 44.
93/11	K.M. van Hee	Systems Engineering: a Formal Approach Part III: Modeling Methods, p. 101.
93/12	K.M. van Hee	Systems Engineering: a Formal Approach Part IV: Analysis Methods, p. 63.
93/13	K.M. van Hee	Systems Engineering: a Formal Approach

		Part V: Specification Language, p. 89.
93/14	J.C.M. Baeten J.A. Bergstra	On Sequential Composition, Action Prefixes and Process Prefix, p. 21.
93/15	J.C.M. Baeten J.A. Bergstra R.N. Bol	A Real-Time Process Logic, p. 31.
93/16	H. Schepers J. Hooman	A Trace-Based Compositional Proof Theory for Fault Tolerant Distributed Systems, p. 27
93/17	D. Alstein P. van der Stok	Hard Real-Time Reliable Multicast in the DEDOS system, p. 19.
93/18	C. Verhoef	A congruence theorem for structured operational semantics with predicates and negative premises, p. 22.
93/19	G-J. Houben	The Design of an Online Help Facility for ExSpect, p.21.
93/20	F.S. de Boer	A Process Algebra of Concurrent Constraint Programming, p. 15.
93/21	M. Codish D. Dams G. Filé M. Bruynooghe	Freeness Analysis for Logic Programs - And Correctness?, p. 24.
93/22	E. Poll	A Typechecker for Bijective Pure Type Systems, p. 28.
93/23	E. de Kogel	Relational Algebra and Equational Proofs, p. 23.
93/24	E. Poll and Paula Severi	Pure Type Systems with Definitions, p. 38.
93/25	H. Schepers and R. Gerth	A Compositional Proof Theory for Fault Tolerant Real-Time Distributed Systems, p. 31.
93/26	W.M.P. van der Aalst	Multi-dimensional Petri nets, p. 25.
93/27	T. Kloks and D. Kratsch	Finding all minimal separators of a graph, p. 11.
93/28	F. Kamareddine and R. Nederpelt	A Semantics for a fine λ-calculus with de Bruijn indices, p. 49.
93/29	R. Post and P. De Bra	GOLD, a Graph Oriented Language for Databases, p. 42.
93/30	J. Deogun T. Kloks D. Kratsch H. Müller	On Vertex Ranking for Permutation and Other Graphs, p. 11.
93/31	W. Körver	Derivation of delay insensitive and speed independent CMOS circuits, using directed commands and production rule sets, p. 40.
93/32	H. ten Eikelder and H. van Geldrop	On the Correctness of some Algorithms to generate Finite Automata for Regular Expressions, p. 17.

93/33	L. Loyens and J. Moonen	ILIAS, a sequential language for parallel matrix computations, p. 20.
93/34	J.C.M. Baeten and J.A. Bergstra	Real Time Process Algebra with Infinitesimals, p.39.
93/35	W. Ferrer and P. Severi	Abstract Reduction and Topology, p. 28.
93/36	J.C.M. Baeten and J.A. Bergstra	Non Interleaving Process Algebra, p. 17.
93/37	J. Brunekreef J-P. Katoen R. Koymans S. Mauw	Design and Analysis of Dynamic Leader Election Protocols in Broadcast Networks, p. 73.
93/38	C. Verhoef	A general conservative extension theorem in process algebra, p. 17.
93/39	W.P.M. Nuijten E.H.L. Aarts D.A.A. van Erp Taalman Kip K.M. van Hee	Job Shop Scheduling by Constraint Satisfaction, p. 22.
93/40	P.D.V. van der Stok M.M.M.P.J. Claessen D. Alstein	A Hierarchical Membership Protocol for Synchronous Distributed Systems, p. 43.
93/41	A. Bijlsma	Temporal operators viewed as predicate transformers, p. 11.
93/42	P.M.P. Rambags	Automatic Verification of Regular Protocols in P/T Nets, p. 23.
93/43	B.W. Watson	A taxomomy of finite automata construction algorithms, p. 87.
93/44	B.W. Watson	A taxonomy of finite automata minimization algorithms, p. 23.
93/45	E.J. Luit J.M.M. Martin	A precise clock synchronization protocol,p.
93/46	T. Kloks D. Kratsch J. Spinrad	Treewidth and Patwidth of Cocomparability graphs of Bounded Dimension, p. 14.
93/47	W. v.d. Aalst P. De Bra G.J. Houben Y. Komatzky	Browsing Semantics in the "Tower" Model, p. 19.
93/48	R. Gerth	Verifying Sequentially Consistent Memory using Interface Refinement, p. 20.

94/01	P. America M. van der Kammen R.P. Nederpelt O.S. van Roosmalen H.C.M. de Swart	The object-oriented paradigm, p. 28.
94/02	F. Kamareddine R.P. Nederpelt	Canonical typing and Π-conversion, p. 51.
94/03	L.B. Hartman K.M. van Hee	Application of Marcov Decision Processe to Search Problems, p. 21.
94/04	J.C.M. Baeten J.A. Bergstra	Graph Isomorphism Models for Non Interleaving Process Algebra, p. 18.
94/05	P. Zhou J. Hooman	Formal Specification and Compositional Verification of an Atomic Broadcast Protocol, p. 22.
94/06	T. Basten T. Kunz J. Black M. Coffin D. Taylor	Time and the Order of Abstract Events in Distributed Computations, p. 29.
94/07	K.R. Apt R. Bol	Logic Programming and Negation: A Survey, p. 62.
94/08	O.S. van Roosmalen	A Hierarchical Diagrammatic Representation of Class Structure, p. 22.
94/09	J.C.M. Baeten J.A. Bergstra	Process Algebra with Partial Choice, p. 16.
94/10	T. verhoeff	The testing Paradigm Applied to Network Structure. p. 31.
94/11	J. Peleska C. Huizing C. Petersohn	A Comparison of Ward & Mellor's Transformation Schema with State- & Activitycharts, p. 30.
94/12	T. Kloks D. Kratsch H. Müller	Dominoes, p. 14.
94/13	R. Seljée	A New Method for Integrity Constraint checking in Deductive Databases, p. 34.
94/14	W. Peremans	Ups and Downs of Type Theory, p. 9.
94/15	R.J.M. Vaessens E.H.L. Aarts J.K. Lenstra	Job Shop Scheduling by Local Search, p. 21.
94/16	R.C. Backhouse H. Doornbos	Mathematical Induction Made Calculational, p. 36.
94/17	S. Mauw M.A. Reniers	An Algebraic Semantics of Basic Message Sequence Charts, p. 9.

94/18	F. Kamareddine R. Nederpelt	Refining Reduction in the Lambda Calculus, p. 15.
94/19	B.W. Watson	The performance of single-keyword and multiple-keyword pattern matching algorithms, p. 46.
94/20	R. Bloo F. Kamareddine R. Nederpelt	Beyond β-Reduction in Church's $\lambda\rightarrow$, p. 22.
94/21	B.W. Watson	An introduction to the Fire engine: A C++ toolkit for Finite automata and Regular Expressions.
94/22	B.W. Watson	The design and implementation of the FIRE engine: A C++ toolkit for Finite automata and regular Expressions.
94/23	S. Mauw and M.A. Reniers	An algebraic semantics of Message Sequence Charts, p. 43.
94/24	D. Dams O. Grumberg R. Gerth	Abstract Interpretation of Reactive Systems: Abstractions Preserving $\forall$CTL*, $\exists$CTL* and CTL*, p. 28.