Abstract. The problem of transforming nondeterministic or concurrent programs involving fairness assumptions into fairness-free schedulers is considered. The availability of such transformations provides an indirect way of proving fair termination. Building upon a general tree-transformation result of Harel, and scheduler-yielding transformations of devised by several researchers, we provide a general way of finding such transformations for a very broad class of fairness criteria. In passing, we explicitly construct a scheduler for Pnueli's rather elusive notion of extreme fairness. Our schedulers are really nondeterministic, and may be termed "cruel" since they leave choices to the program, but immediately thereafter cause it to abort if it happens to make the wrong choice.

Abstract. This paper provides a common framework to study transformations of structures ranging from all kinds of graphs to relational data structures. Transformations of structures can be used as derivations of graphs in the sense of graph grammars, as update of relations in the sense of relational data bases, or even as operations on data structures in the sense of abstract data types. The main aim of the paper is to construct parallel and concurrent transformations from given sequential ones and to study sequentializability properties of complex transformations. The main results are three fundamental theorems concerning parallelism, concurrency and decomposition of transformations of structures. On one hand these results can be considered as a contribution to the study of concurrency in graph grammars and on the other hand as a formal framework for consistent concurrent update of relational structures.

Abstract. For, aside from the fact that the concepts occurring in this calculus possess an objective importance and are in these times almost indispensable in any scientific discussion, the calculus of relations has an intrinsic charm and beauty which makes it a source of intellectual delight to all who become acquainted with it. ALFRED TARSKI [10].

Abstract. Many linear-time graph algorithms using depth-first search have been invented. We propose simplified versions of two such algorithms, for computing a bipolar orientation or st-numbering of an undirected graph and for finding all feedback vertices of a directed graph.

9 (2) 1986

Abstract. A mechanism is introduced to describe priorities in *ACP*, the algebra of communicating processes, whereby some actions have priority over others in a non-deterministic choice (or sum). This mechanism can be used to model the working of interrupts in a distributed system. This is illustrated in an extensive example.

Keywords: concurrency, process algebra, interrupts, priorities.

Abstract. Three small programming languages are introduced: a procedural one, an applicative one and a functional one (here functional is always meant in the sense of "functional in the style of Backus"). For each of them mathematical (denotational) and operational semantics are defined. Then the similarities and differences between these three languages with respect to mathematical semantics, operational semantics, syntactic and notational properties are discussed in detail. Specific attention is paid to the notion of program variable

9 (3) 1986

MODULAR REAL-TIME TRELLIS AUTOMATA

Anton Černý, and Jozef Gruska *253-282*

Abstract. A new type of nonhomogeneous real time trellis automata, the so-called modular trellis automata, is introduced and various results concerning their normal forms, power, simulations, and decision problems are shown. Modular trellis automata are a mathematical abstraction, in the form of a recognizer, of an intuitive notion of an array of simple processors assembled in a simple and modular way. Distribution of processors in a real-time trellis automaton forms a two-dimensional structure called trellis. Basic characterizations and properties of modular trellises are summarized and modularity of various special trellises - regular, product, self-embedding, and self-overlapping - is investigated.

ALMOST CONTEXT-FREE LANGUAGES

Michal P. Chytil *283-322*

Abstract. We define a superclass of the class of context-free languages, denoted ACFL (almost context-free languages) and construct an infinite sequence of non-context-free languages of decreasing complexity, belonging to ACFL. The languages in ACFL share many important properties of context-free languages. For example, a slight generalization of pumping lemma for context-free languages holds also for languages in ACFL. Similarly, some of the questions decidable for context-free languages are decidable in ACFL and start to be undecidable immediately outside the class. ACFL is a full AFL and contains an infinite strictly decreasing chain of full AFLs.

Keywords: almost context-free language, context-free language, one-way auxiliary pushdown automaton, pumping lemma, abstract family of languages, complexity of computations.

ON THE COMPLEXITY OF DEADLOCK RECOVERY

Joseph Y.-T. Leung, and Burkhard Monien *323-342*

Abstract. We consider the computational complexity of finding an optimal deadlock recovery. It is known that for an arbitrary number of resource types the problem is NP-hard even when the total cost of deadlocked jobs and the total number of resource units are "small" relative to the number of deadlocked jobs. It is also known that for one resource type the problem is NP-hard when the total cost of deadlocked jobs and the total number of resource units are "large" relative to the number of deadlocked jobs. In this paper we show that for one resource type the problem is solvable in polynomial time when the total cost of deadlocked jobs or the total number of resource units is "small" relative to the number of deadlocked jobs. For fixed $m \geq 2$ resource types, we show that the problem is solvable in polynomial time when the total number of resource units is "small" relative to the number of deadlocked jobs. On the other hand, when the total number of resource units is "large", the problem becomes NP-hard even when the total cost of deadlocked jobs is "small" relative to the number of deadlocked jobs. The results in the paper, together with previous known ones, give a complete delineation of the complexity of this problem under various assumptions of the input parameters.

ALGEBRAIC THEORIES, DATA TYPES, AND CONTROL CONSTRUCTS

Eric G. Wagner *343-370*

Abstract. The aim of this paper is to model recursive types, equational types, and elementary programming control constructs (such as conditionals and while-do) in one, comparatively simple,

algebraic framework, that can be used for theoretical studies and as a basis for data type and program specification. To this end we introduce a new kind of algebraic theory, the RV-theory. We give simple examples of the use of such theories for data type specification. We provide a mathematical semantics for these specifications that extends the initial algebra semantics for equational specification to include recursive types.

9 (4) 1986

PROVING MONITORS REVISITED: A FIRST STEP TOWARDS VERIFYING OBJECT ORIENTED SYSTEMS

Rob Gerth, and W.P. De Roever *371-400*

Abstract. An axiomatic characterization of monitors. based on assumption-commitment style reasoning, is given that is sound and (relatively) complete. This characterization is based on the fundamental notions of *cooperation* and *interference*, but does not use them as second order notions. The cooperation test was originally conceived to capture the proof theoretical analogue of distributed message passing between disjoint processes, as opposed to the interference freedom test, being the proof theoretical analogue of concurrency based on interference by jointly shared variables. Since then, the cooperation test has been applied to characterize synchronous communication in Hoare's Communicating Sequential Processes, Ichbia's Ada, and Wirth's Modula-2, supported by soundness and completeness proofs. An overview is given of the rationale underlying this characterization, culminating in the development of proof systems for a new monitor based programming language for concurrency (*Communicating Modules*, CM) which combines distributed message passing between processes with interference through local variables of a process which are shared between its sub-processes. As such this is a first step towards the formal verification of object oriented systems. In this context, we also show how the method, traditionally cauched in terms of proof outlines, can be rendered syntax directed in a precise and formal way. In a separate paper, the proof system has been shown to be sound and (relatively) complete.

A COMPLETE PROOF SYSTEM FOR SCCS WITH MODAL ASSERTIONS

Glynn Winskel *401-420*

NOTIONS OF REALIZABLE NON-SEQUENTIAL PROCESSES

C. Fernandez, M. Nielsen, and P.S. Thiagarajan *421-454*

Abstract. This paper presents some results on non-sequential processes using the language of net theory. The results are concerned with the relationship between various formalizations of the intuition that the causality relation enforced by a process should be in some sense "finitely realizable". The formalizations proposed are of very different flavours, based on notions of observability, approximability, state space covering and discreteness.

COMPUTATIONS IN COORDINATED PAIR SYSTEMS

A. Ehrenfeucht, H.J. Hoogeboom, and G. Rozenberg *455-480*

Abstract. *Selective substitution grammars* provide a rather general framework for the grammatically oriented formal language theory (see, e.g., [R1], [K], and [KR]). They were generalized in [R2] to *coordinated table selective substitution systems (cts systems)* which provide a convenient unifying framework for both *grammars and machines (automata)*. The present paper investigates *coordinated pair (cp) systems* which form a subclass of cts systems corresponding in a very natural way to push-down automata: thus cp systems generate context-free languages (all and only). This paper investigates the structure of computations in cp systems.

ITERATION THEOREMS FOR DETERMINISTIC FAMILIES OF LANGUAGES

Michael A. Harrison *481-508*

Abstract. In this paper, we consider the problem of finding iteration theorems for various subfamilies of deterministic languages. Because deterministic languages are constrained in their generation, it is not possible to merely "pump substrings" as in the general context free case. We lay out, in detail, a collection of techniques for proving theorems of this type for deterministic context free languages.