Technical Reports Librarian
Department of Computer Science
Columbia University
New York, NY 10027

[CUCS-045-91] Optimal scheduling of task tree on constant dimensional architectures.

Xiangdong Yu.

One important aspect of parallel/distributed computation is the processing of a set of statically given tasks on some specific architectures. We consider tasks whose precedence relationship has the form of a tree and architectures that have constant dimensional grids like structures. In this paper we give an efficient scheme for mapping tasks to processors and for routing results in the architecture such that the whole set of tasks can be executed in optimal time. Many researches can be found in literatures on similar problems whereas to our knowledge, the result in this paper is the first to achieve optimal time and optimal processor time product in a model that considers both task dependency and communication overhead, in addition to computation cost.

[CUCS-046-91] A formal proof to a basic fact in pseudorandom generation.

Xiangdong Yu.

In the definition of cryptographically strong pseudorandom generator, $1/poly(n)$ is used as the criterion for the difference of probabilities with which testers correctly predict the next bit of a generator. Intuitively, a super tester, such as an exponential time tester, can predict with maximum accuracy and thus can make the maximum difference from predicting by blind guessing. We prove, this maximum difference is always greater than $1/p(n)$ for $p(n) = 2(n + 1)$, therefore no generator can be strong for a super tester. In other words, pseudorandom is really "pseudo" with respect to this criterion.

[CUCS-047-91] Knowledge representation and reasoning with definitional taxonomies.

Robert A. Weida.

We provide a detailed overview of knowledge representation issues in general and terminological knowledge representation in particular. Terminological knowledge representation, which originated with KL-ONE, is an object-centered approach in the tradition of semantic networks and frames. Terminological systems share three distinguishing characteristics: (1) They are intended to support the definition of conceptual terms comprising a "terminology" and to facilitate reasoning about such terms. As such, they are explicitly distinguished from assertional systems which make statements of fact based on some terminology. (2) Their concepts are arranged in a taxonomy so that the attributes of a concept apply to its descendants without exception. Thus, the proper location of any concept within the taxonomy can be uniquely determined from the concept's definition by an automatic process known as classification. (3) They restrict the expressiveness of their language to achieve relatively efficient performance. We first survey important general issues in the field of knowledge representation, consider the semantics of concepts and their interrelationship, and examine the intertwined notions of taxonomy and inheritance. After discussing classification, we present a number of implemented terminological systems in detail, along with several hybrid systems which couple terminological and assertional reasoning components. We conclude by assessing the current state of the art in terminological knowledge representation.

[CUCS-041-91] Machine learning in molecular biology sequence analysis.

Philip K. Chan.

To investigate how human characteristics are inherited, molecular biologists have been analyzing chemical sequences from DNA, RNA, and proteins. To facilitate this process, sequence analysis knowledge has been encoded in computer programs. However, translating human knowledge to programs is known to be problematic. Machine Learning techniques allow these systems to be generated automatically. This article discusses the application of learning techniques to various analysis tasks. It is shown that the learned systems constructed to date are often more accurate than human-designed systems. Moreover, learning can form plausible new hypotheses, which potentially lead to discovering new knowledge.

[CUCS-042-91] On the set lcs and set-set lcs problems.

G. Chen B. Wang, , and K. Park.

We consider two generalizations of the longest common subsequence (LCS) problem: the Set LCS problem and the Set-Set LCS problem. We present algorithms for the two problems that are faster than the previous ones by Hirschberg and Larmore.

[CUCS-043-91] Fully dynamic dictionary matching.

Z. Galil, R. Giancarlo, and K. Park.

We consider the dynamic dictionary matching problem. We are given a set of pattern strings (the dictionary) that can change over time; that is, we can insert a new pattern into the dictionary or delete a pattern from it. Moreover, given a text string, we must be able to find all occurrences of any pattern of the dictionary in the text. Let D_0 be the empty dictionary. We present an algorithm that performs any sequence of the following operations in the given time bounds:

[1] (1) $insert(p, D_{i-1})$: Insert pattern $p[1,m]$ into the dictionary D_{i-1}. D_i is the dictionary after the operation. The time complexity is $O(m \log |D_i|)$.

[2] (2) $delete(p, D_{i-1})$: Delete pattern $p[1,m]$ from the dictionary D_{i-1}. D_i is the dictionary after the operation. The time complexity is $O(m \log |D_{i-1}|)$.

[3] (3) $search(t, D_i)$: Search text $t[1,n]$ for all occurrences of the patterns of dictionary D_i. The time complexity is $O((n + tocc) \log |D_i|)$, where $tocc$ is the total number of occurrences of patterns in the text. Our algorithm improves in several respects Amir and Farach's recent solution for the dynamic dictionary matching problem.

[CUCS-044-91] A formal characterization of epsilon serializability.

Calton Pu Krithi Ramamritham.

Epsilon Serializability (ESR) is a generalization of classic serializability (SR). ESR allows some limited amount of inconsistency in transaction processing (TP), through an interface called epsilon-transactions (ETs). For example, some query ETs may view inconsistent data due to non-SR interleaving with concurrent updates. In this paper, we restrict our attention to the situation where query-only ETs run concurrently with *consistent* update transactions that are SR without the ETs. This paper presents a *formal characterization* of ESR and ETs. Using the ACTA framework, the first part of this characterization formally expresses the inter-transaction conflicts that are recognized by ESR and, through that, defines ESR, analogous to the manner in which conflict-based serializability is defined. The second part of the paper is devoted to deriving expressions for: (1) the *inconsistency* in the values of data – arising from ongoing updates, (2) the inconsistency of the results of a query – arising from the inconsistency of the data read in order to process the query, and (3) the inconsistency exported by an update ET – arising from ongoing queries reading uncommitted data produced by the update ET. These expressions are used to determine the preconditions that ET operations have to satisfy in order to maintain the limits on the inconsistency in the data read by query ETs, the inconsistency exported by update ETs, and the inconsistency in the results of queries. This determination suggests possible mechanisms that can be used to realize ESR.

to store and retrieve information about feature specifications and test scripts in the context of a large software development project.

[CUCS-037-91] Towards automatic vehicle navigation.

Il-Pyung Park.

Driving a car is a task that requires intense concentration of the driver. This task becomes even more formidable when the driver has to figure out how to get from one place to another. For example, driving along an unfamiliar area while at the same time looking at a paper map is not easy if not dangerous. With the development of computer and communication technologies, it is becoming possible to drive efficiently and safely. This paper surveys various research areas and technologies involved in development of systems that help drivers to navigate. These include human psychology, dead reckoning, map matching, proximity beacons, satellites, LORAN-C systems, digital maps, driver interface, efficient data structures, and algorithms. The ultimate goal of all these, is of course, automatic vehicle navigation, where the car drives by itself.

[CUCS-038-91] Fuf user manual - version 5.0.

Michael Elhadad.

This report is the user manual for FUF version 5.0, a natural language generator program that uses the technique of unification grammars. The program is composed of two main modules: a unifier and a linearizer. The unifier takes as input a semantic description of the text to be generated and a unification grammar, and produces as output a rich syntactic description of the text. The linearizer interprets this syntactic description and produces an English sentence. This manual includes a detailed presentation of the technique of unification grammars and a reference manual for the current implementation (FUF 5.0). Version 5.0 now includes novel techniques in the unification allowing the specification of types and the expression of complete information. It also allows for procedural unification and supports sophisticated forms of control.

[CUCS-039-91] Empirical studies on the disambiguation of cue phrases.

Julia Hirschberg and Diane Litman.

Cue phrases are linguistic expressions such as @i(now) and @i(well) that function as explicit indicators of the structure of a discourse. For example, @i(now) may signal the beginning of a subtopic or a return to a previous topic, while @i(well) may mark subsequent material as a response to prior material, or as an explanatory comment. However, while cue phrases @b(may) convey discourse structure, each also has one or more alternate uses. While @i(incidentally) may be used @b(SENTENTIALLY) as an adverbial, for example, the @b(DISCOURSE) use initiates a digression. The question of how speakers and hearers distinguish between discourse and sentential uses of cue phrases is rarely addressed in discourse studies. This paper reports results of several empirical studies of discourse and sentential uses of cue phrases, in which both text-based and prosodic features were examined for disambiguating power. Based on these studies, we propose that discourse versus sentential usage may be distinguished by intonational features, specifically, @b(PITCH) @b(ACCENT) and @b(INTONATIONAL) @b(PHRASING). We identify a prosodic model that characterizes these distinctions. We associate this model with features identifiable from text analysis, including orthography and part-of-speech, to permit the application of our findings to the generation of appropriate intonational features for discourse and sentential uses of cue phrases in synthetic speech.

[CUCS-040-91] Parallel dynamic programming.

Z. Galil and K. Park.

We study the parallel computation of dynamic programming. There are five important dynamic programming problems which have wide application, and that have been studied extensively in sequential computation. Two problems among the five have fast parallel algorithms; almost no work has been done for parallelizing the other three. We give fast parallel algorithms for four of these problems, including the three. We use two well-known methods as general paradigms for developing parallel algorithms. Combined with various techniques, they lead to a number of new results. At the heart of our new results is a processor reduction technique, which enables us to solve all four problems with a fewer number of operations than that needed for computing a matrix closure.

[CUCS-032-91] Marvel 3.0 administrator's manual.

Programming System Laboratory.

This manual is intended for system/project administrators of Marvel 3.0. It covers various aspects of managing a Marvel environment, including installation of the system, administrator built-in commands, and mainly, how to design and write a Marvel environment using Marvel Strategy Language (MSL) and Shell Envelope Language (SEL).

[CUCS-033-91] Marvel 3.0 user's manual.

Programming System Laboratory.

This manual is intended for Marvel end-users, who want to learn how to use Marvel in a specific environment set-up by an administrator. It includes a tutorial, and explanations on the user-interface, messages, and the various built-in commands.

[CUCS-034-91] Real-time visual servoing.

A. Timcenko P. Allen, B. Yoshimi.

This paper describes a new real-time tracking algorithm in conjunction with a predictive filter to allow real-time visual servoing of a robotic arm that is tracking a moving object. The sytem consists of two calibrated (but unregistered) cameras that provide images to a real-time, pipelined-parallel optic-flow algorithm that can robustly compute potic-flow and calculate the 3-D position of a moving object at approximately 5 Hz rates. These 3-D positions of the moving object serve as input to a predictive kinematic control algorithm that uses an alpha-beta-gamma filter to update the position of a robotic arm tracking the moving object. Experimental results are presented for the tracking of a moving model train in a variety of different trajectories.

[CUCS-035-91] Automated tracking and grasping of a moving object with a robotic hand-eye system.

P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman.

Most robotic grasping tasks assume a stationary or fixed object. In this paper, we explore the requirements for tracking and grasping a moving object. The focus of our work is to achieve a high level of interaction between a real-time vision system capable of tracking moving objects in 3-D and a robot arm equipped with a dexterous hand that can be used pick up a moving object. We are interested in exploring the interplay of hand-eye coordination for dynamic grasping tasks such as grasping of parts on a moving conveyor system, assembly of articulated parts or for grasping from a mobile robotic system. Coordination between an organism's sensing modalities and motor control system is a hallmark of intelligent behavior, and we are pursuing the goal of building an integrated sensing and actuation system that can operate in dynamic as opposed to static environments. The system we have built addresses three distinct problems in robotic hand-eye coordination for grasping moving objects: fast computation of 3-D motion parameters from vision, predictive control of a moving robotic arm to track a moving object, and grasp planning. The system is able to operate at approximately human arm movement rates, and we present experimental results in which a moving model train is tracked, stably grasped, and picked up by the system. The algorithms we have developed that relate sensing to actuation are quite general and applicable to a variety of complex robotic tasks that require visual feedback for arm and hand control.

[CUCS-036-91] Taxonomic plan reasoning.

Premkumar T. Devanbu Diane J. Litman.

CLASP (ClAssification of Scenarios and Plans) is a knowledge representation system that extends the notion of subsumption from frame-based languages to plans. The CLASP representation language provides description forming operators that specify temporal and conditional relationships between actions represented in CLASSIC (a current subsumption-based knowledge representation language). CLASP supports subsumption inferences between plan concepts and other plan concepts, as well as between plan concepts and plan instances. These inferences support the automatic creation of a plan taxonomy. Subsumption in CLASP builds on term subsumption in CLASSIC and illustrates how term subsumption can be exploited to serve special needs. In particular, the CLASP algorithms for plan subsumption integrate work in automata theory with work in term subsumption. We are using CLASP

Gail E. Kaiser, Israel Ben-Shaul, and Steven S. Popovich.

Our goal was to implement the activity structures model defined by Software Design & Analysis on top of the MARVEL environment kernel. This involved further design of the activity structures process definition language and enaction model as well as translation and run-time support in terms of facilities provided by MARVEL. The result is an elegant declarative control language for multi-user software processes, with data and activities defined as classes and rules in the previously existing MARVEL Strategy Language. Semantics-based concurrency control is provided by a combination of the MARVEL kernel's lock and transaction managers and the send/receive synchronization primitives of the activity structures model.

[CUCS-028-91] Service interface and replica management algorithm for mobile file system clients.

Carl D. Tait and Dan Duchamp.

Portable computers are now common, a fact that raises the possibility that file service clients might move on a regular basis. This new development requires re-thinking some features of distributed file system design. We argue that existing approaches to file replica management would not cope well with the likely behavior of mobile clients, and we present our solution: a lazy "server-based" update operation. This operation facilitates fast, scalable, and highly fault-tolerant implementations of both read and write operations in the usual case. To cope with the weak semantics of the update operation, we propose a new file system service interface that allows applications to opt for "UNIX semantics" by use of a slower, less fault-tolerant read operation.

[CUCS-029-91] An architecture for dynamic reconfiguration in a distributed object-based programming language.

Brent Hailpern and Gail E. Kaiser.

Distributed applications ideally allow reconfiguration while the application is running, but changes are usually limited to adding new client and server processes and changing the bindings among such processes. In some application domains, such as real-time financial services, it is necessary to support finer grained reconfiguration at the level of entities smaller than processes, but desirable to reduce the overhead associated with ad hoc dynamic storage allocation. We present a scheme for special cases of fine-grained dynamic reconfiguration sufficient for our application domain and show how it can be used for practical changes. We introduce new language concepts to apply this scheme in the context of an object-based programming language that supports shared data in a distributed environment.

[CUCS-030-91] Real-time portfolio management and automatic extensions.

Tushar M. Patel.

Voluminous amounts of rapidly changing data in financial markets create a challenging problem for portfolio managers attempting to exploit such changes to achieve their investment objectives. The SPLENDORS real time portfolio management system, built using the PROFIT language, is such a system. Changes in security prices are monitored in a real-time fashion and individual portfolios can be optimized by taking advantage of these changes within the constraints imposed by the investment philosophy of the investor. SPLENDORS allows great flexibility in creating portfolio managers for the sophisticated C programmer and considerable ease of programming for the PROFIT programmer. Additionally, the non-programming financial analyst may also create portfolio managers from parts of a predefined library without any knowledge of programming at all. Extensions to SPLENDORS made by the end user are available immediately without bringing the system down for re-compilation.

[CUCS-031-91] Navigating the meldc: The meldc user's manual.

Howard Gershen and Erik Hilsdale.

MeldC is an attempt to create a computer language that is a watershed of developments in the fields of object-oriented programming, parallel programming, and distributed applications. This easy-to-understand manual provides readers with the details of how to write programs in MeldC, in addition to plain-English explanations of some of the theory (e.g., object/class hierarchies, race condition problems, and remote-file access problems) behind MeldC's features. Illustrated. Commented sample programs and BNF included.

units. Solutions to these problems make it possible to construct semantics-based editors for use by teams of programmers developing or maintaining large software systems. Multi-user semantics-based editors improve software productivity by reducing communication costs and snafus. The objectives of an incremental attribute evaluation algorithm for multiple asynchronous changes are that (a) all attributes of the derivation tree have correct values when evaluation terminates, and (b) the cost of evaluating attributes necessary to reestablish a correctly attributed derivation tree is minimized. We present a family of algorithms that differ in how they balance the tradeoff between algorithm efficiency and expressiveness of the attribute grammar. This is important because multi-user editors seem a practical basis for many areas of computer-supported cooperative work, not just programming. Different application areas may have distinct definitions of efficiency, and may impose different requirements on the expressiveness of the attribute grammar. The characteristics of the application domain can then be used to select the most efficient strategy for each particular editor. To address the second problem, we define an extension of classical attribute grammars that allows the specification of interface consistency checking for programs composed of many modules. Classical attribute grammars can specify the static semantics of monolithic programs or modules, but not inter-module semantics; the latter was done in the past using @i[ad hoc] techniques. Extended attribute grammars support programming-in-the-large constructs found in real programming languages, including textual inclusion, multiple kinds of modular units and nested modular units. We discuss attribute evaluation in the context of programming-in-the-large, particularly the separation of concerns between the local evaluator for each modular unit and the global evaluator that propagates attribute flows across module boundaries. The result is a uniform approach to formal specification of both intra-module and inter-module static semantic properties, with the ability to use attribute evaluation algorithms to carry out a complete static semantic analysis of a multi-module program.

[CUCS-024-91] Execution autonomy in distributed transaction processing.

Calton Pu and Avraham Leff.

We study the feasibility of execution autonomy in systems with asynchronous transaction processing based on epsilon-serializability (ESR). The abstract correctness criteria defined by ESR are implemented by techniques such as asynchronous divergence control and asynchronous consistency restoration. Concrete application examples in a distributed environment, such as banking, are described in order to illustrate the advantages of using ESR to support execution autonomy.

[CUCS-025-91] Performance comparison of sender-active and receiver-active mutual data serving.

Calton Pu, Danilo Florissi, Patricia Soares, , Philip S. Yu, and Kun-Lung Wu.

In a distributed system, data servers (file systems and databases) can easily become bottlenecks. We propose to offload data access requests from overloaded data servers to nodes that are idle or less busy, called mutual servers. In addition to server and client local caches, frequently accessed data are cached in the main memory of mutual servers, thus improving the data access time in the system. We evaluate several data propagation strategies among data servers and mutual servers. Simulation results show significant performance improvement for both Sender-Active and Receiver-Active protocols. For a wide range of cache sizes and load conditions, the active policies achieve a mean response time half that of pure client/server.

[CUCS-026-91] An architectural framework for object management systems.

Steven S. Popovich.

Much research has been done in the last decade in the closely related areas of object-oriented programming languages and databases. Both areas now seem to be working toward a common end, that of an @i(object management system), or OMS. An OMS is constructed similarly to an OODB but provides a general purpose concurrent object oriented programming language as well, sharing the object base with the OODB query facilities. In this paper, we will define several different types of object systems (object servers, persistent OOPL's, OODB's and OMS's) in terms of their interfaces and capabilities. We will examine the distinguishing features and general architecture of systems of each type in the light of a general model of OMS architecture.

[CUCS-027-91] Implementing activity structures process modeling on top of the marvel environment kernel.

[CUCS-019-91] Dynamic data structures for graphs.

Giuseppe Francesco Italiano.

Ph.D. Thesis.

We consider dynamic algorithms and dynamic data structures for various graph problems, such as edge connectivity, vertex connectivity and shortest paths. We first show how to maintain efficiently a solution to the shortest-path problem during edge insertions and edge-cost decreases. We then present the first sublinear-time algorithms known for maintaining the 2-edge-connected components of a general graph and the 2-vertex-connected components of a planar graph during edge and vertex insertions and deletions. Finally, we introduce a fast algorithm for maintaining the 3-edge-connected components of a graph during edge insertions only. The solutions of all the above problems share many common features and combine a variety of graph properties, novel algorithmic techniques and new data structures.

[CUCS-020-91] Control in functional unification grammars for text generation.

Michael Elhadad and Jacques Robin.

Standard Functional Unification Grammars (FUGs) provide a structurally guided top-down control regime for text generation that is not appropriate for handling non-structural and dynamic constraints. We introduce two control tools that we have implemented for FUGs to address these limitations: *bk-class*, a tool to limit search by using a form of dependency-directed backtracking and *external*, a co-routine mechanism allowing a FUG to cooperate with dynamic constraint sources. We show how these tools complement the top-down regime of FUGs to enhance lexical choice.".

[CUCS-021-91] Mpd: A multiprocessor debugger (m.s. thesis).

Krish Ponamgi.

MpD is a multiprocessor C debugger designed for multithreaded applications running under the Mach operating system. MpD is built on top of gdb, an existing sequential debugger. The MpD layer utilizes the modeling languages Data Path Expressions developed by Hseush and Kaiser to provide a rich set of commands to trace sequential and parallel execution of a program. Associated with each DPE are actions that allow access to useful trace variables and I/O facilities. DPEs are useful for describing sequential and concurrent patterns of events to be verified during execution. The patterns include conditions such as synchronizations, race conditions, and wrongly classified sequential/concurrent behavior. We show in this thesis Data Path Expressions are a viable language for multiprocessor debuggers.

[CUCS-022-91] Rule chaining in MARVEL: Dynamic binding of parameters.

"George T. Heineman, Gail E. Kaiser, Naser S. Barghouti, and Israel Z. Ben-Shaul.

MARVEL is a rule-based development environment (RBDE) that assists in the development of software projects. MARVEL encapsulates each software development activity in a rule that specifies the condition for invoking the activity and its effects on the components of the project under development. These components are abstracted as objects and stored in a persistent object database. Each rule applies to a specific class of objects, which is specified as the parameter of the rule. Firing a rule entails binding its formal parameter to a specific object. If the rule changes the object in such a way that the conditions of other rules become satisfied, these other rules are automatically fired. A problem arises in this forward chaining model when the classes of the objects manipulated by the rules are different. MARVEL has to determine which object to bind to the parameter of each rule in the chain, based on the object manipulated by the original rule that initiated the chain. We describe a heuristic approach for solving this problem in the current MARVEL implementation and introduce an algorithmic approach that does better.

[CUCS-023-91] Incremental attribute evaluation for multi-user semantics-based editors.

Josephine Micallef.

Ph.D. Thesis.

This thesis addresses two fundamental problems associated with performing incremental attribute evaluation in multi-user editors based on the attribute grammar formalism: (1) multiple asynchronous modifications of the attributed derivation tree, and (2) segmentation of the tree into separate modular

[CUCS-013-91] Debugging multi-threaded programs with mpd.

M. Krish Ponamgi, Wenwey Hseush, and Gail E. Kaiser.

Also available in IEEE Software, May 1991.

We have developed a Multi-processor Debugger, MpD, based on Data Path Expressions and Predecessor Automata. Data Path Expressions (DPEs) is a debugger command language for the programmer to express both sequential and parallel patterns of events. Predecessor Automata (PAs) are automatically generated recognizers for DPEs. The programmer specifies DPEs to represent event patterns that may happen during program execution, and the PAs analyze the combined event streams collected from the executing threads to determine whether or not each DPE was actually matched.

[CUCS-014-91] Extending a tool integration language.

Mark A. Gisi and Gail E. Kaiser.

The MARVEL environment supports rule-based automation of software processes. MARVEL invokes external tools to carry out steps in a software process. One of the major objectives of this research is to invoke external tools to carry out steps in a software process without modifying the tools. This is achieved by encapsulating tools in *envelopes*, designed to abstract the details of a tool from the MARVEL kernel, thereby providing a "black box" interface. Initially we used the UNIX shell script language to write envelopes. However, due to several limitations of the shell language the black box abstraction could not be fully supported. We describe these limitations and discuss how we extended the shell language to obtain a new envelope language that fully supports the black box abstraction.

[CUCS-015-91] Using ipip for remote bridging at the ip-level.

John Ioannidis and Gerald Q. Maguire Jr.

We describe the use of the IPIP encapsulation protocol to tunnel traffic between two physically separated network segments that must have the same IP network and subnet numbers.

[CUCS-016-91] Fully dynamic algorithms for 2-edge connectivity.

Zvi Galil and Giuseppe F. Italiano.

We study the problem of maintaining the 2-edge-connected components of a graph undergoing repeated dynamic modifications, such as edge insertions and edge deletions. We show how to test at any time whether two vertices belong to the same 2-edge-connected component, and how to insert and delete an edge in $O(m^{2/3})$ time in the worst case, where m is the current number of edges in the graph. This answers a question posed by Westbrook and Tarjan [WT89]. For planar graphs, we present algorithms that support all these operations in $O(\sqrt{n \log\log n}\,)$ worst-case time each, where n is the total number of vertices in the graph.

[CUCS-017-91] Maintaining the 3-edge-connected components of a graph on-line.

Zvi Galil and Giuseppe F. Italiano.

We study the problem of maintaining the 3-edge-connected components of a graph undergoing repeated dynamic modifications, such as edge and vertex insertions. We show how to answer whether two vertices belong to the same 3-edge-connected component of a connected graph that is undergoing only edge insertions. Any sequence of q query and updates on an n-vertex graph can be performed in $O((n+q)\alpha(q,n))$ time.

[CUCS-018-91] Maintaining biconnected components of dynamic planar graphs.

Zvi Galil and Giuseppe F. Italiano.

We present algorithms for maintaining the biconnected components of a planar graph undergoing repeated dynamic modifications, such as insertions and deletions of edges and vertices. We show how to test at any time whether two vertices belong to the same biconnected component, and how to insert and delete an edge in $O(n^{2/3})$ time in the worst case, where n is the number of vertices in the graph. The data structure supports also insertions of new vertices and deletions of disconnected vertices in the same time bounds. This is the first sublinear-time algorithm known for this problem.

we describe the asynchronous execution of transactions for greater performance and availability. The solution is based on epsilon-serializability (ESR) that models temporary inconsistency explicitly. The database maintains serializable consistency, but applications that can tolerate a bounded amount of inconsistency may gain additional concurrency and availability. Efficient replica control and divergence control methods make ESR-based asynchronous TP attractive. The similarities between these methods and traditional concurrency control methods make them clearly feasible. These positive steps in the evolution towards a truly heterogeneous and autonomous TP system are just the beginning. We discuss some of the challenges in both areas and in their combination.

[CUCS-010-91] An object management system for multi-user programming environments.

Israel Ben-Shaul.

Multi-user design environments impose specific requirements on data management that differ substantially from those imposed by traditional business applications, due to the difference in the type and duration of frequent operations on the data, and the structure of and the access patterns to the data. In particular, it is desired to enable sharing and exchange of information for collaborative work while still maintaining data-consistency. The object-oriented paradigm seems like a promising approach in terms of data modeling but imposes difficulties in terms of data management in multi-user systems. Two models are proposed here, both are based on the object-oriented data model: A centralized model, in which all data management is done at a single node, and a distributed model, in which the data management is done in multiple nodes, each of which is structured according to the centralized model specifications. An implementation of the centralized model for @marvel, a multi-user rule-based software development environment, is given. The emphasis in the centralized model is on "componentization" of the system into independent, replaceable components while still coping with the inherent complexity of the data. The emphasis in the distributed model is on exploiting the object-oriented paradigm into distribution, and it concentrates on data partitioning and dynamic reconfiguration. The emphasis in the implementation of the model is on specific problems related to rule-based programming environments, and on practical implementation solutions.

[CUCS-011-91] The splendors real time portfolio management system (extended abstract).

Tushar M. Patel and Gail E. Kaiser.

Splendors is a sample real time portfolio management application built using Profit. Splendors provides a packaged library of common components as well as facilities for populating additional libraries. The libraries make it possible for financial analysts and brokers, usually non-programmers, to construct their own portfolio managers. The end-users carry out a simple, menu-based parameterization of selected components in order to incorporate their specific expertise and investment philosophies. The resulting portfolio managers are then added to the Splendors system, without the need to bring down the application for recompilation and restart.

[CUCS-012-91] Dynamic restructuring of transactions.

Gail E. Kaiser and Calton Pu.

Also available in Database Transaction Models for Advanced Applications, Morgan Kaufmann, 1991.

Open-ended activities are characterized by uncertain duration, unpredictable developments, and interactions with other concurrent activities. Like other database applications, they require consistent concurrent access and fault-tolerance, but their unconventional characteristics are incompatible with the conventional database mechanisms of concurrency and failure atomicity. We present the split-transaction and join-transaction operations for *restructuring* in-progress transactions, as an approach to consistent concurrent access and fault-tolerance for open-ended activities. Split-transaction divides an on-going transaction into two or more transactions that are serializable with respect to each other and all other transactions, and each of the new transactions is later committed or aborted independently of the others. Join-transaction merges two or more transactions that are serializable with respect to each other into a single transaction as if they had always been part of the same transaction, and all their work is now committed or aborted together. Split-transaction is useful for committing some work early or dividing on-going work among several co-workers. Join-transaction allows to hand over results to a co-worker to integrate into his or her own ongoing task. The transaction manager enforces that the new transaction(s) will in fact be serializable, and does not permit the split or join otherwise.

[CUCS-005-91] A lock-free multiprocessor os kernel.

H. Massalin and C. Pu.

Typical shared-memory multiprocessor OS kernels use interlocking, implemented as spin-locks or waiting semaphores. We have implemented a complete multiprocessor OS kernel (including threads, virtual memory, and I/O including a window system and a file system) using only lock-free synchronization methods based on Compare-and-Swap. Lock-free synchronization avoids many serious problems caused by locks: considerable overhead, concurrency bottlenecks, deadlocks, and priority inversion in real-time scheduling. Measured numbers show the low overhead of our implementation, competitive with user-level thread management systems.

[CUCS-006-91] IP-based protocols for mobile internetworking.

John Ioannidis, Dan Duchamp, and Gerald Q. Maguire Jr.

We consider the problem of providing network access to hosts whose physical location changes with time. Such hosts cannot depend on traditional forms of network connectivity and routing because their location, and hence the route to reach them, cannot be deduced from their IP address. We present protocols that seamlessly integrate mobile hosts into the current IP networking infrastructure. They are primarily targeted at supporting a campus environment with mobile computers, but also extend gracefully to accomodate hosts moving between different networks. The key feature is the dependence on ancillary machines to track the location of the mobile hosts. Our protocols are designed to react quickly to changing topologies, to scale well, and not to place an overwhelming burden on the network.

[CUCS-007-91] Implementing the visibility constraint in the "mvp" machine vision planning system using the acis 3-d solid modeler.

Steven Abrams and Konstantinos Tarabanis.

In this paper we describe a technique by Tarabanis and Tsai from their *"MVP"* (Machine Vision Planning) system to analytically determine the locus of points from which a given polygonal target can be viewed without obstruction by occluding polyhedral objects. In addition, a description of the implementation of this visibility algorithm in the ACIS 3-D Solid Geometric Modeling System is given. Preliminary experimental results are presented in which this visibility constraint is combined with the depth of field, field of view, and resolution constraints of the *"MVP"* system in order to generate both *feasible* and *optimal* (at least locally) camera position, pose, and optical settings.

[CUCS-008-91] Inferring constraints from multiple snapshots.

David Kurlander and Steven Feiner.

Many graphics tasks, such as the manipulation of graphical objects, and the construction of user-interface widgets, can be facilitated by geometric constraints. However, the difficulty of specify-ing constraints by traditional methods forms a barrier to their widespread use. In order to make constraints easier to declare, we have developed a method of specifying constraints implicitly, through multiple examples. Snapshots are taken of an initial scene configuration, and one or more additional snapshots are taken after the scene has been edited into other valid configurations. The constraints that are satisfied in all the snapshots are then applied to the scene objects. We discuss an efficient algorithm for inferring constraints from multiple snapshots. The algorithm has been incorporated into the Chimera editor, and several examples of its use are discussed.

[CUCS-009-91] Heterogeneous and autonomous transaction processing.

C. Pu, A. Leff, and S.W.F. Chen.

Transaction processing (TP) systems have been evolving towards greater distribution, heterogeneity, and autonomy. In the heterogeneity and autonomy dimensions, we present concrete and independent steps in this evolution starting from solutions in classic distributed TP. For cooperative heterogeneous TP, we describe the Superdatabase architecture, which provides transaction atomicity over heterogeneous component databases. Superdatabase shows that if a component database supports local recovery and concurrency control, necessary ingredients in classic distributed TP, then it can become part of a cooperative heterogeneous TP system. The feasibility of superdatabase architecture is demonstrated by the Supernova prototype implementation in the Harmony project. For autonomous TP execution,

[CUCS-001-91] Measurement of camelot performance over the internet.

Calton Pu and Fred Korz.

We have measured the end-to-end performance of Camelot distributed transactions running over the Internet. Such measurements are difficult because of the uncontrollable routing and queueing instabilities in the Internet. We apply the Layered Refinement methodology to acquire and analyze data. Results show interesting characteristics of both the Internet and the Camelot.

[CUCS-002-91] Divergence control for epsilon-serializability.

Kun-Lung Wu, Philip S. Yu, and Calton Pu.

This paper presents divergence control methods for epsilon-serializability (ESR) in centralized databases. ESR alleviates the strictness of serializability (SR) in transaction processing by allowing for limited inconsistency. The bounded inconsistency is automatically maintained by divergence control (DC) methods in a way similar to SR is maintained by concurrency control (CC) mechanisms. However, DC for ESR allows more concurrency than CC for SR. In this paper, we design DC methods in such a way that read-only transactions need not be serializable with other update transactions, but update transactions must be serializable among themselves. ESR is both feasible and widely applicable. In this paper, we first demonstrate the feasibility of ESR by showing the designs of representative DC methods. The total number of non-serializable (non-SR), conflicting operations that a transaction can encounter is used as an inconsistency specification. DC methods are designed by systematically enhancing CC algorithms in two stages: extension and relaxation. In the extension stage, a CC algorithm is analyzed to locate the places where it identifies non-SR conflicts of database operations. In the relaxation stage, the non-SR conflicts are relaxed to allow for controlled inconsistency. Three representative DC methods are presented: two-phase locking, timestamp ordering, and optimistic approaches. For each DC method, we show that the ESR properties are preserved. We then demonstrate the applicability of ESR by presenting the designs of DC methods using other inconsistency specifications. The DC designs based on non-SR, conflicting operations are extended, with only minor changes, to implement most known inconsistency specifications, such as absolute value, age, and total number of data items changed.

[CUCS-003-91] Efficient communication in optical computers.

Mihály Geréb-Graus and Thanasis Tsantilas.

We study the problem of interprocessor communication in a parallel computer model, suggested by recent advances in optical technology. The units of the computer (processors with local memory) communicate with each other by transmitting messages. If two or more processors try to send a message to the same processor no transmission is succesful and retransmission must occur. For any $h = \Omega(\log n \log\log n)$, we exhibit a simple randomized algorithm that realizes arbitrary h-relations on an n-node optical computer in time $O(h)$. In an h-relation, each processor has to transmit at most h messages and each processor is to receive at most h messages. Our algorithm is *pure* in the sense that no information aside from actual packets is transmitted between processors. We also believe that our algorithm is practical.

[CUCS-004-91] Adaptive remote paging for mobile computers.

Bill N. Schilit and Dan Duchamp.

There is a strong trend toward the production of small "notebook" computers. The small size of portable computers places inherent limits on their storage capacity, making remote paging desirable or necessary. Further, mobile computers can "walk away from" their servers, increasing load on network routing resources and/or breaking network connections altogether. Therefore, it is desirable to allow client-server matchups to be made dynamically and to vary over time, so that a client might always be connected to nearby servers. Accordingly, we have built a self-organizing paging service that adapts to changes in locale and that stores pages in remote memory if possible. We show empirically that there is no performance penalty for using our paging facility instead of a local disk. This suggests that portable computers need neither a hard disk nor an excessive amount of RAM, provided that they will operate in environments in which remote storage is plentiful. These are important facts because both a hard disk and large amounts of RAM are undesirable characteristics for very small portable computers.

Attn:Technical Reports Librarian
Department of Computer Science
Columbia University
New York, NY 10027

Columbia University in the City of New York
Department of Computer Science
New York, NY 10027
(212) 854-2736
tech-reports@cs.columbia.edu

TECHNICAL REPORTS LIST

The Computer Science Department at Columbia University is pleased to announce the enclosed list of technical reports and abstracts for 1991.

If you wish to order any of these reports, please complete the form below, fold so that the return address is properly positioned, and affix postage.

A NOTE TO INTERNET USERS: many Columbia Computer Science technical reports are available by anonymous FTP from host `cs.columbia.edu` directory `pub/reports`. All reports are stored in compressed postscript format.

Technical Report Numbers

1. _____	13. _____	25. _____	37. _____
2. _____	14. _____	26. _____	38. _____
3. _____	15. _____	27. _____	39. _____
4. _____	16. _____	28. _____	40. _____
5. _____	17. _____	29. _____	41. _____
6. _____	18. _____	30. _____	42. _____
7. _____	19. _____	31. _____	43. _____
8. _____	20. _____	32. _____	44. _____
9. _____	21. _____	33. _____	45. _____
10. _____	22. _____	34. _____	46. _____
11. _____	23. _____	35. _____	47. _____
12. _____	24. _____	36. _____	

Name _____

Address _____

City _____ State _____ Zip _____