The Use of Semantic Information for Control of a Complex Routing Tool*

Dr Mike Brown Dr Zahir Moosa Dr Nick Filer

Department of Computer Science, The University of Manchester
Oxford Road, Manchester, UK, M13 9PL

Abstract

To handle increasingly complex design data, CAD tools are becoming more specialised and complex and hence, more difficult to use. This paper describes an interactive system that helps a designer to use such tools more effectively. A concrete example is provided based on a powerful and innovative routing tool.

1 Introduction

The research described in this paper is part of a large-scale project developing a CAD framework (namely, the Jessi-Common-Frame). CAD frameworks provide a means of integrating the many tools and data resources that are typically used in a CAD environment under a single management system. In this way, a framework helps reduce the complexity of the environment, for example, by providing a consistent user interface for the invocation of tools and access of data. However, scope remains for the development for more sophisticated tools to aid the designer in better exploiting the facilities at their disposal. In particular, a system is described in this paper that provides guidance to a designer in the use of a complex CAD tool.

The layout of the paper is as follows. Section 2 introduces the general concept of a Design Consultancy Support (DCS) system and explains how such a system can benefit a designer in the use of complex CAD tools. Section 3 introduces the TRACKER routing tool, an example of a complex CAD tool. Section 4 gives a worked example of the type of advice that can enable a user to more effectively exploit a tool. Finally, section 5 provides a summary of the paper and describes the future development of this research.

2 User Assistance in Complex Design Environments

In this section, the need is explained for automated assistance in complex design environments.

2.1 Design Consultancy Support

The growing complexity of design environments is generating a new type of problem for designers. The diversity of tools and data sources available, as well as the increased complexity of CAD tools, means that it can no longer be assumed that a designer is fully aware of the facilities available to them and how to best exploit these facilities to complete design tasks [9].

DCS systems have the role of over-coming this problem by aiding the designer in the over-all management of the design process. Typically, Knowledge-Based System (KBS) technology is employed to provide facilities such as; focussing of the user's activities [9], support for decision points in the design process [6, 8], collating diverse information sources [6, 2] and automated logging of the design process [1].

Ordinarily, DCS support takes place at a level upwards of the invocation CAD tools; assistance is given for the selection of an appropriate tool for a particular design task but it is then the user's responsibility to exploit that tool. In contrast, it is argued in this paper that DCS technology can also be profitably employed to guide the use of complex tools.

2.2 The Role of DCS in Tool Usage

There are several problems associated with tool usage that may be overcome using a DCS system. Firstly, when a designer is faced with an unfamiliar tool, they expend valuable time in learning to use it. The difficulty lies in mapping the requirements of the designer, in terms of the task they wish to perform, to the primitive operations of the chosen tool; a single design task generally requires a complex combination of many sub-tool operations. A potential benefit of a DCS system lies in the acceleration of this learning process.

The problem of learning to use a tool *effectively* goes beyond a simple understanding of the tool's functionality. It is necessary to understand the circumstances for which the various facilities offered are most appropriate and what are the limitations of each individual facilities. This information is valuable expertise and making it widely available through the use of a DCS system has great potential benefit. As will be shown, it can lead to a general reduction in

*Part of Esprit Project 7364 (Jessi-Common-Frame).

the time taken to complete a design task using the tool. In addition, it may lead to the avoidance of misuse of the tool (i.e. the failure to satisfactorily complete the design task, for example due to a failure to optimally set tool control parameters [5]). Though the principles described in this paper apply to any CAD tool, DCS assistance offers most benefit for non-interactive tools with potentially long invocation times, such as simulation and layout tools, where there is limited scope for rapid recovery from any mistake on the user's part.

2.3 An Architecture for a Tool Support System

Supporting the designer in the use of an unfamiliar and/or complex CAD tool embodies many of the traditional aspects of DCS. In particular, the behaviour of the tool can be represented as a design flow using the same type of formalism (e.g. [8, 11, 6]) used to capture higher-level design behaviour. In this way, the task of determining the appropriate tool usage for a particular design becomes the task of determining the appropriate path through the tool's design flow. In particular, a design flow explicitly represents the optional decision points in a tool's behaviour that can be resolved by a user (either through direct interaction or indirectly through setting control parameters) and hence explicitly indicates where DCS assistance can be invoked. Design flows also provides a basis by which the various information sources required to resolve decision points can be organised [3]; the representation of each source will be associated with a particular position in the flow.

One type of information required is meta-level knowledge, such as the amount of time allotted for a designer to complete a given task. A DCS system may also have access to "deep" semantics associated with, but not explicitly represented within, the design data [12]. Another information type is pragmatic knowledge about the tool itself, such as estimates of the execution time for each operation and documented problems. Many other types of relevant information can be identified and the example of section 4 gives an illustration of how a diversity information can be used in the generation of sensible DCS advice.

A minimum requirements can now be established for a DCS system to support tool usage. The DCS system should; provide the user with a transparent design flow model of the tool's behaviour, allow the user to generate paths through this design flow (for example by parameter selection) and provide a generic framework by which a user can gain access to appropriate information by which to resolve decision points in the design flow. This leads to a general system architecture as shown in figure 1.

3 A Complex CAD Tool

In this section, TRACKER (an innovative and powerful pcb routing tool) is described. TRACKER is the basis for the concrete example of the application of DCS techniques at the level of tool usage.

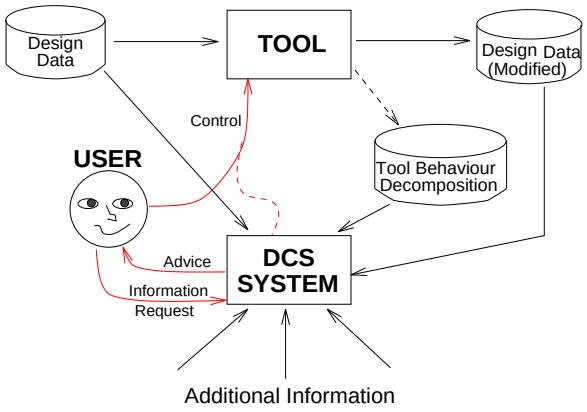

Figure 1: The General Role of DCS in Tool Support

3.1 Novel Approaches to Routing

Lee's algorithm [10] is the most popular and general purpose routing technique. Nets are routed in a predetermined order and cost parameters can be used to govern the general nature of the connections.

A fundamental problem with Lee's algorithm is its fixed routing order. This frequently causes earlier routed nets to unnecessarily block later nets [13]. To overcome this problem, TRACKER embodies a novel "rip-up and reroute" strategy [14, 13] whereby nets are permitted to cross each other during routing and intersecting nets are continuously redirected, such that the process tends towards a state of zero intersections and minimum failed nets. In order to force this dynamic strategy to converge to a solution, the *simulated annealing* [7] optimisation technique is employed. Initially, to avoid local optima, this technique allows routing to generate worse states. The acceptability of worse states is controlled non-deterministically and decreases as the algorithm *'cools'*; in this way the algorithm converges towards a solution (at the *'freezing point'*).

The simulated annealing approach has been shown [14, 13] to significantly reduce the number of routing failures (with respect to a conventional Lee's algorithm approach) for a wide range of real design boards. The practical result of the research is a powerful tool (i.e. TRACKER) which supports a number of possible routing techniques including traditional serial implementations of Lee's algorithm and the simulated annealing approach. Never-the-less, the behaviour of the tool is difficult to understand and often unpredictable to control [14] which detracts from its usability. Enabling the potential power of TRACKER to be better harnessed is the goal for the exemplar DCS application described in this paper.

3.2 An Overview of the TRACKER Tool

In this section, the hierarchy of design flows that represents the TRACKER routing tool (as modelled by the DCS prototype) will be outlined. A top-level design flow for the

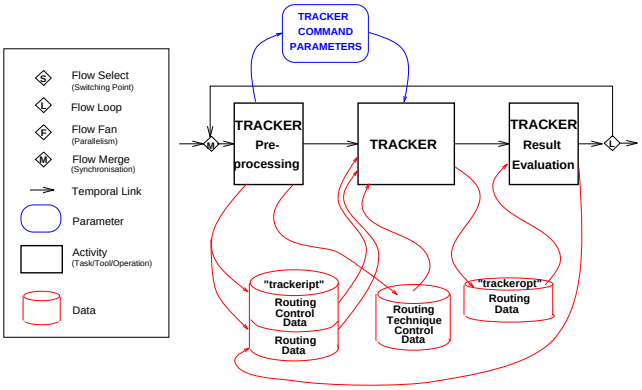

Figure 2: A Top-Level view of TRACKER

tool is given in figure 2. The various data resources associated with the tool are illustrated in conjunction with the temporal ordering of activities. Figure 2 is an instantiation of a generally applicable design flow for tool invocation. In this design flow, the task of preprocessing always precedes the actual running of a tool and the task of result evaluation (with the potential to loop back and re-run the tool) always follows the completion of the tool.

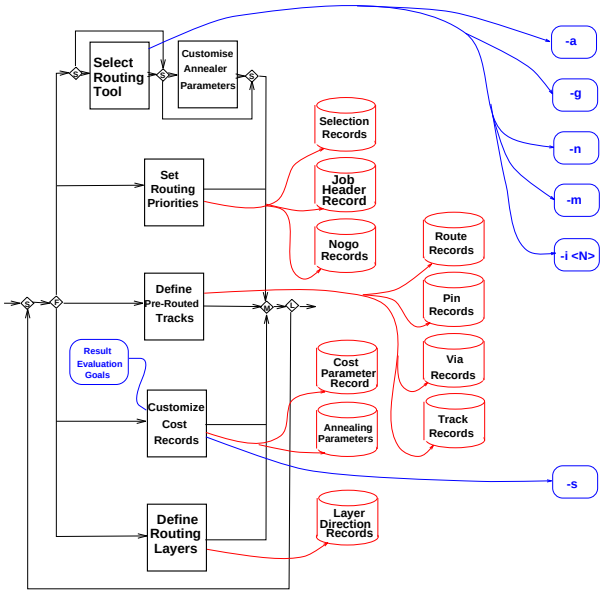

Figure 3: TRACKER Preprocessing

Preprocessing generally involves a collection of relatively independent tasks that are used to configure various aspects of a tool so as to influence its behaviour. This influence is usually manifest through the editing of control record files or through the setting of command line parameters. The preprocessing tasks that are applicable to TRACKER are shown in the design flow of figure 3.

The design flow representing the behaviour of a tool

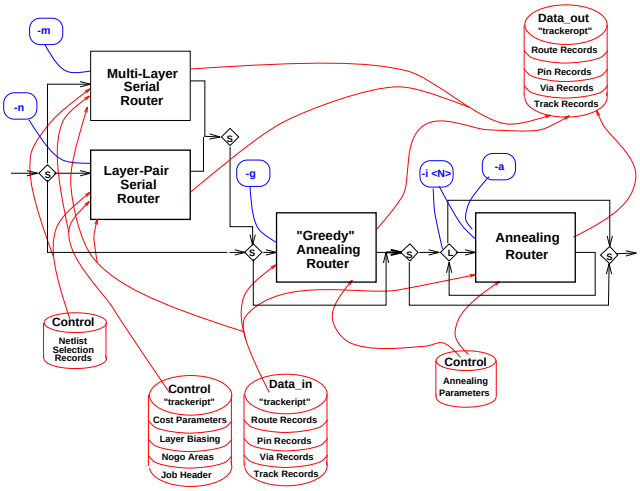

Figure 4: TRACKER internals

should explicitly represent the major operations carried out by that tool. The design flow of figure 4 details TRACKER's internal behaviour. The various available techniques for performing routing are shown. This design flow also shows precisely which parts of TRACKER's operation are selected by its control parameters, which control records are relevant to each component and the use of design data resources.

The existing DCS prototype [2] uses an interactive window-based interface to present design flows to the user. In this way, the behaviour of an otherwise opaque and non-interactive tool is made transparent. This in itself is a useful aid for a user who is unfamiliar with TRACKER. However, additional supporting information is required so as to enable a designer to use TRACKER effectively. Such information is used during the preprocessing activity in order to configure TRACKER to best suit a particular board. The next section gives a sample of the types of information that are relevant.

3.3 The Use of Semantic Information to Control TRACKER

In general, a prerequisite to enable the correct configuration of a tool is to have knowledge of the characteristics of the various components/operations offered by that tool. An informal description of the four main components of TRACKER is given below:

- **Layer-pair serial router.** This routes by only considering pairs of routing layers with opposing directional biases. Additional pairs of layers are used (if available) when routing failures are encountered. Generally, this component has a short execution time but is of low power (i.e. will produced many failed nets for difficult input data).

- **Multi-layer serial router.** This is the default router. It routes on all available layers simultaneously. It generally distributes tracks more evenly than the layer-pair router. Again this component is fast but of low power.
- **Annealing router.** This employs the technique of simulated annealing described in section 3.1. The performance of this router is critically determined by the setting of various control variables, such as the *temperature decrement factor* and the *start temperature* [14]. The speed of performance is also governed by the initial state of the board. For complex boards, it is generally beneficial to generate this state by preceeding the annealing router with one of the other routing techniques, typically the multi-layer serial router. The annealing router produces significantly fewer errors on hard to route boards than the serial routers, but at the expense of time (typically having an execution time 10-100 times longer).
- **"Greedy" annealing router.** This is a version of the annealing router that converges to a solution more rapidly, though with less chance of reaching an optimal solution.

As well as knowledge that matches semantic features of the design data to tool option characteristics (hence enabling the selection of the appropriate option), meta-level strategic knowledge for use of a tool is generally required. For example, in some circumstances it is beneficial to use TRACKER iteratively, for instance when it is advantageous to route a board by stages. As an example of this, on boards where "hot-spots" can be predetermined, an effective routing strategy is to first route those nets that lie within these densely inter-connected areas then, as a second stage, to route the rest of the board. Similarly, sometimes certain predetermined signals, such as clock signals, may require special treatment during routing so should be routed separately (see section 4).

This section has given little more than a taste of the types of knowledge required in order to control CAD tools generally and TRACKER specifically. Evaluating the strategy for using the tool on a given board is a complex task and an automated system for achieving this may have to rely on a heterogeneous set of data resources [3]. As is true for DCS in general [15], it is likely that a variety of reasoning agents and a variety of representation formalisms will be required to provide full, automated support for tool usage. Because of the complexity of the task, the emphasis for the current DCS prototype is on allowing the designer to bring their own expertise and intuition to bear on the problem through the storage and presentation of appropriate information types [2].

4 A Worked Example

As an illustration of the potential intricacy of using a CAD tool, a worked example, based on TRACKER, is described in this section.

4.1 Problem Description

The example is based on a real memory board that was custom designed at the University of Manchester. The board involves four tracking layers and is problematic to route for a number of reasons [4]:

- There is a high inter-connectivity between the components on the board.
- The density of nets is spread unevenly such that several areas exist which have disproportionately high number of nets within them.
- The board's nets include sensitive clock signals that must be routed so as to minimise signal impedance (i.e. short, straight connections).

For arguments sake, its is also assumed that the time for completing the routing task is limited, as might typically be the case in a commercial design situation. Hence the goal for the DCS system is to suggest a customisation of the TRACKER tool that will allow for the generation of the least number of routing failures in the time avaliable (i.e. maximise goal satisfaction).

4.2 DCS Support for TRACKER

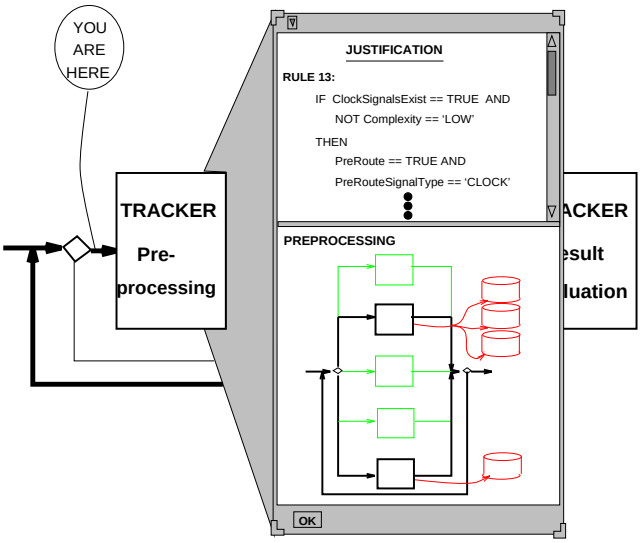

Figure 5: Elaborating a Design Flow Component

The first task for the DCS system is to establish a general strategy for use of the tool. This can be done using rule-based diagnosis, where the "symptoms" are characteristics of the design data and the "prognoses" are design flow selection. For example, assuming the presence of clock signals can be detected in the design data, strategic knowledge for routing dictates that the corresponding nets should be

routed first. The DCS system can present this information to the user by highlighting the appropriate preprocessing tasks that need to be carried out to realise this goal, as well as making transparent its reasoning (i.e. the appropriate production rules). The presentation of this information to the user is shown in figure 5.

Similarly, diagnostic rules can be used to configure a tool (i.e. select the appropriate parts of its behavioural design flow) to best suit the input data. For example, once the clock signals have been routed, the DCS system may aid the designer in configuring TRACKER to route the remaining nets. Based on semantic information indicating the high complexity of the board, a sensible DCS proposal (based on the characteristics of each TRACKER component) for is to use the annealing or greedy router preceeded by the multi-layer router. Again this information can be conveyed by highlighting the chosen design flow and allowing the designer to inspect the underlying reasoning.

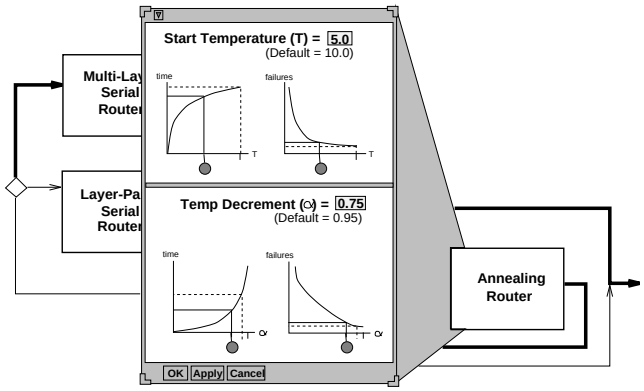

Figure 6: Help for Parameter Setting

There are other types of information other than that which can be formalised in a symbolic representation (e.g. diagnostic rules) that can be of benefit when using a tool. In particular, statistical information can be useful for fine-tuning a tools performance [5]. For this research, a particularly important class of statistical relationship is the effect of changing a control input of a tool on a measure of tool performance with respect to specific goal criteria. This is illustrated below.

Because of the tight deadline for the example routing scenario, a designer may need to know how the simulated annealing router can be configured so as to produce a faster convergence than default without an excessive increase in failure rate. To help in this way the DCS system can present to the user statistical relationships between various annealing control parameters and the execution time and relative failure rate for that router (see figure 6). This information is unlikely to be known by even expert designers. Never-the-less, it is require if a successful customisation of TRACKER is to be performed. This requires adjustment of the control parameters from their default values so as to generate a reduction in routing time without significantly increasing the failure rate. The graphs of figure 6 can be used by a designer to intuitively make this compromise. The next section demonstrates the improved results that can be achieved in this way.

Description	Failures	Time (CPU Secs)
Multi-Layer Alone (Default Tracker)	135	168
Multi-Layer + Greedy Router	27	11871
Multi-Layer + Default Annealing Router	16	19582
Multi-Layer + Customised Annealing Router	17	7944

Table 1: Routing Results

4.3 The Effect on Performance

Table 1 represents results achieved using various set-ups of TRACKER to complete the routing of the memory board. Row 1 shows the results achieved using the default mode for TRACKER (i.e. no information input). Though very rapid, the high failure rate produced (135 un-routed nets) is unacceptable which justifies the need for the user to perform some preprocessing for the TRACKER tool.

Rows 2 and 3 represent two sensible alternatives based on the guidance of a DCS system for component selection alone. In both cases, a relatively low error rate is achieved (27 and 16 respectively) but the time to complete routing is high. Note, as is expected, the greedy router performs faster than the annealing router but produces a slightly higher failure rate.

Row 4 represents the results achieved if, in addition to component selection, the DCS system also provides the user with statistical information on how to customise the control parameters of a particular component. In this case, the start temperature and temperature decrement factor for the simulated annealing router are set in accordance with figure 6. The number of failures produced (17) is comparable to that of rows 3 but the execution time is less than half that of the default annealer (a saving of over 3 hours). It can be concluded that the DCS advice detailed in the previous section does indeed lead to a use of TRACKER that best suits the example problem description.

5 Conclusions

In this paper, it has been argued that complex CAD tools can be made more usable by a wide range of designers if accompanied by a DCS system that presents a variety of support information types to a user as well as performing some automated diagnosis with respect to the circumstances of a particular tool invocation. This principle has been demonstrated by showing how a prototype DCS system can provide sensible advice concerning the

use of a novel routing tool, leading to an improvement in tool performance as well as improved user understanding of the tool's behaviour.

One of the key requirements for this type of DCS system is to formally represent the behaviour and data requirements of the tool as a design flow. This provides a framework for organised representation of the various other types of information required for controlling tool usage as it makes explicit at what point in the operation of the tool a particular data item has an effect.

A potential area for future investigation is the symbiotic relationship that may exist between the proposed type of DCS application and development of the tool itself. It should be possible to adapt tool development methodologies so as to generate, as a by-product, an explicit representation of the tool's behaviour in a design flow formalism. From the opposite perspective, using a DCS system to monitor the use of a tool could provide a means for generating useful information to feed back into the tool development cycle. As an example, a DCS system could automatically collate statistics as to what operational components of a tool are used with greatest frequency and/or success.

The current state of this research is that an extensive period of knowledge acquisition based on the TRACKER tool has been completed and an initial prototype DCS system has been constructed [3, 2]. This system allows a user to interactively navigate the tool's design flows and to then automatically invoke the tool. Conversion of the elicited knowledge into a production rule system and a detailed evaluation of the results of automated execution of the TRACKER tool is one aspect of the current research effort. In addition, a component is being developed for the automated extraction and presentation of the types of statistical information described in section 4.2. The long term goal is the development of a generic methodology for providing knowledge-based assistance for effective tool usage.

References

[1] W Allen, D Rosenthal, and K Fiduk. The MCC CAD Framework Methodology Management System. In *Proc. of the DAC-91*, pages 694–698, San Fransisco, 1991. ACM.

[2] M Brown, N P Filer, and Z Moosa. Design and Development of the MOWGLI-II Prototype Design Flow Manager. Technical Report JCF/MAN/105-01/28-May-94, ESPRIT Project 7364 : JESSI-Common-Frame, 1994.

[3] M Brown, N P Filer, and Z Moosa. Knowledge Required for Design Flow Management. Technical Report JCF/MAN/104-01/28-May-94, ESPRIT Project 7364 : JESSI-Common-Frame, 1994.

[4] D Edwards, A MacIntosh, Z Moosa, and F Hoyle. University of Manchester PCB Software. internal document, August 1993.

[5] S Fujita, M Otsubo, and M Watanabe. An Intelligent Control Shell for CAD Tools. In *Proc. of CAIA-94*, pages 16–22, San Antonio, Texas, 1994.

[6] F Kemper, A Scherer, M Straube, W Wilkes, and G Schlageter. A Knowledge-Based Design Consultant: Model and Architecture. In *Proc. of the 25th Hawaii Int. Conf. on System Science*, pages 542–550. IEEE Computer Society Press., 1992.

[7] S Kirkpatrick, C D Gelatt, and M P Vecchi. Optimization by Simulated Annealing. *Science*, 220, 1983.

[8] C Kocourek. A Petri Net Based Design Decision Support System. In *Proceedings of the International Conference Applied Modelling and Simulation*, 1993.

[9] E Kwee-Christoph, B Eschermann, and D Schmid. A Design Consultant to Support CAD Tool Usage. In *Proc. of ASIC-93*, New York, 1993.

[10] C Y Lee. An Algorithm for Path Connections and its Application. *I.R.E Transactions on Electronic Computers*, pages 346–365, 1961.

[11] A McKnight. Flexible Design Methodology Management. In *Proc. of the IEE Colloquium on Design Management Enviroments in CAD*, pages 1–6, 1991.

[12] Salvador Mir. *Heuristic Reasoning for an Automatic Commonsense Understanding of Logic Electronic Design Specifications*. PhD thesis, The University of Manchester, 1993.

[13] Z Moosa, M Brown, and D Edwards. An Application of Simulated Annealing to Maze Routing. In *Proc. of EURO-DAC '94*, Grenoble, 1994. IEEE-CS Press.

[14] Zahir Moosa. *On Improving Maze Routing Algorithms*. PhD thesis, The University of Manchester, 1993.

[15] M Straube, W Wilkes, and G Schlageter. HANDICAP - A System for Design Consulting. In *Proceedings of EDAC-94*, Paris, 1994.