The Third International Conference on the Practical Applications of Prolog, Paris, 3rd-6th April 1995, pp. 135-144.

Constraint-based Maintenance Scheduling on an Electric Power-Distribution Network

Tom Creemers[1], Lluís Ros Giralt[1], Jordi Riera[1], Carles Ferrarons[2], Josep Roca[2], Xavier Corbella[2]

[1] Institut de Cibernètica (UPC / CSIC)
Diagonal, 647, 2ª planta
08028 Barcelona, Spain
Telephone: + 34 3 401 66 53
Fax: +34 3 401 66 05
email: creemers,ros,riera@ic.upc.es

[2] ENHER, S.A.
Passeig de Gràcia, 132
08008 Barcelona, Spain
Telephone: + 34 3 415 50 00
Fax: + 34 3 415 75 72

Abstract: The exploitation of a power-distribution network involves the scheduling of multiple maintenance and unforeseen repair tasks. The main resource is a network subject to topological, economical and electric constraints. A line section being maintained needs to be isolated from the rest of the network by opening all surrounding switches. This, in turn, would leave other areas of the network de-energized, which is unacceptable in most cases. Hence, these areas have to get their supply via some alternative way, i.e., service needs being restored closing switches connecting to an energized part of the network, taking into account overloading of branches, energy losses, and the cost of the necessary switching operations. In case tasks are carried out in the same area, switching operations might be shared among them. In some cases a valid network reconfiguration might not even exist. Finally, typical scheduling constraints have to be met: resources of limited availability (manpower, vehicles, etc.), due dates, priority relations, etc. We present *PLANETS*, a prototype scheduler based on CHIP, generating near-optimal schedules for the tasks to be carried out in one week making sure that, at any moment in time, the network is appropriately reconfigured to guarantee power supply to all consumers. The minimization criterion involves lateness, priority and total cost of the necessary switching operations.

Keywords: scheduling, combinatorial optimization, resources, constraints, power-distribution network reconfiguration, service restoration.

1. Introduction

1.1 Problem statement

Imagine the following scene at the dispatching center of an electric company. The *planning engineer* has just ordered the send-off of a vehicle to close switch number *6576* of the *distribution network*. The purpose: to keep on supplying electricity to some consumers affected by maintenance of an upstream line section. However, the planning engineer soon realizes that, two days before, switch number *6576* had already been closed for some time, and that by properly advancing the repair tasks in the schedule, the extra displacement of the vehicle to the switch could have been saved. At the dispatching center, these and other similar situations arise frequently in the daily exploitation of the network. The planning engineer, who, among other assignments, has to lead and coordinate all maintenance tasks on the distribution network, is at the same time responsible for finding those solutions that cost least to the company. Every week, he has to design the *maintenance schedule* for the following one, optimally distributing in time all foreseen maintenance tasks. Moreover, he has to figure out how to reconfigure the distribution network to be able to carry out this schedule.

The preparation of such schedule is not trivial. Certainly, the engineer is facing a complex decision problem since he needs to consider lots of constraints and assign values to many variables. Indeed:

- In order to carry out every foreseen maintenance task, it is first of all necessary to *isolate* the affected line section by opening all nearest surrounding switches. Next, the engineer has to decide through which of the many possible alternative paths of the network, he will keep on energizing all consumers affected by a maintenance task. He has to *reconfigure* the network while keeping it radial, i.e., there may not exist any closed loops, and ensuring that current limits of all lines are not exceeded.

- The number of disconnected *consumers* has to be minimized. The disconnection of a customer is only justified in certain cases where there does not exist any reconfiguration of the network that can maintain power supply. The amount of unconsumed energy translates directly into profit loss for the company.

- He has to achieve optimal usage of the company's limited resources involved in maintenance activities. These are basically: equipment, mobile technical staff, transportable generators, distribution operators, and the distribution network itself. By properly constructing the schedule this is possible.

- He has to consider the dynamic behaviour of the *energy demand* along the week. Thus, for example, a maintenance task involving the disconnection of a main line carrying high current values, cannot be performed at peak hours. The overload produced in neighbouring lines as a consequence of the reconfiguration could cause significant damage.

- Additionally, he has to meet temporal constraints such as priorities, *due dates* and, for some tasks, *a priori* fixed dates.

Finally, if we bear in mind the huge size of the electric networks dealt with (they can easily contain 2000 nodes and 800 switches), we understand the combinatorial complexity of the problem the planning engineer is periodically faced with.

1.2 Previous approaches

Presently, electric companies are spending a lot of human resources in dealing with this scheduling problem. However, the search for electrically and economically acceptable schedules is generally a slow, manual, and iterative process mainly based on the experience of the planning engineer himself.

Many research has been carried out in the specific field of optimal reconfiguration of power networks for both load balancing and energy-loss reduction. In general, these methods allow, given a set of network areas, of which is known *precisely* at which moment they will be disconnected (either due to routine maintenance or sudden failure), to determine the optimal reconfiguration of the still operative network that maintains supply to all consumers. Typical goals are the minimization of either energy losses (due to Joule's effect) or overloads in line sections. In this field several approaches have been taken. We cite some.

In [Shirmohammadi 92], a fast heuristic method is developed that solves the reconfiguration problem. To minimize the overall overload, the algorithm first meshes the network closing all switches and, afterwards, it iteratively opens the least charged switch, until a radial configuration is reached. In [Aoki 88], a different approach is taken based on operations-research techniques. The reconfiguration problem is modelled as a *subset-sum problem* taking advantage of several well known existing techniques to solve it. Rule-based systems have also been used extensively to tackle reconfiguration problems ([Tsai 93], [Liu 88]). In these systems, the empirical knowledge of human dispatching operators is modelled by means of rules. [Wong 87] describes a method to allocate load among substations based on logic-programming techniques. In this work, a depth-first search was used to find ways of adding load without violating operating constraints. Constraints are used passively, in a generate-and-test fashion. Finally, one can also find connectionist approaches in [Jung 93] or genetic search algorithms in [Nara 92].

All these techniques are very useful for reconfiguration purposes. However, their main drawback is that, when being used to design the weekly maintenance schedule, they can only treat the reconfiguration problem once the operator has fixed "by hand" the starting and finishing dates of all outage zones under maintenance. In this way, once a particular temporal distribution of outages is chosen, it often happens that consumers are left unsupplied as a result of the disconnection of multiple line sections for maintenance purposes. Obviously, due to the tremendous combinatorial possibilities of the network topology, it is impossible for a dispatching operator to inspect all possible orderings of tasks that avoid this problem, and choose the best one. Moreover, these methods do not allow to reduce redundant switching-operations. The system described in this paper tries to remedy these two shortcomings by treating both the generation of a maintenance schedule and the finding of the associated optimal reconfiguration plan, as one *single* global optimization problem. As far as we know, it is also the first tool using the constraint-logic-programming paradigm for optimal network reconfiguration.

2. The Approach: Constraint Logic Programming

2.1 Related work

- **Scheduling**

Since the beginning of the 1980s a lot of work has been done in constraint-based scheduling. An historical perspective is outlined in [Le Pape 94]. The development of tools like CHIP [Van Hentenryck 89] and ILOG Solver [Puget 94], especially when enhanced with cumulative resource constraints [Aggoun 92], formed the basis for the implementation of precise, efficient, flexible and extensible industrial scheduling and resource allocation systems.

- **Electrical Engineering**

Most of the work done on the use of constraints in the electrical engineering field involves *digital circuits*. Examples are *verification* [Simonis 88], *diagnosis* [Simonis 87], *synthesis* [Simonis 90] and *test-pattern generation* [Simonis 89]. These examples highlight the ease and flexibility with which digital circuits can be modelled in CHIP and its efficiency in their problem solving. Their relevance to distribution-network reconfiguration lies in the fact that to the latter is, to a large extent, a "switching" problem. A large industrial circuit-design application is described in [Filkorn 91].

Less work is described on *analog circuits*, although, as pointed out in [Heintze 91], the field is rich in problems that are complex and largely numeric, and provided the driving examples for the pioneering work on constraint programming. The authors demonstrate the use of real arithmetic constraints in CLP(?) on a number of small electrical engineering problems. The models used are both general and rich, allowing the representation of a wide range of electric circuits. The main drawback, especially for large circuits, seems the overall presence of *non-linear arithmetic constraints* which are delayed until they become linear. Hybrid, *digitally controlled analog systems* are described in [Nerode 93].

2.2 An integrated solution to two problems

The above examples demonstrate that constraint-logic-programming languages are suitable for modelling and solving problems on electric circuits. In the particular case of power distribution, we have to deal with extremely large networks of a few thousands of nodes. Many of the branches are *switches*, introducing a large number of *0-1* variables. An acceptable network state, defined as the collection of the states of all of its switches, is however always "radial". Moreover, there must always, apart from some exceptions, exist a closed path between any consumer node and a root node (a transformer), i.e., a consumer must always receive power supply.

It is also clear from past experiences that any "scheduling problem" is a possible candidate for a constraint-based approach. We show that the electric and topological constraints which delimit the possible states of the network, cut additionally the search space of all possible

schedules. In other words, the states of certain switches at a particular moment in time, constrain the execution of maintenance jobs in certain parts of the network at that same time. Conversely, a (partial) solution for the scheduling problem cuts the space of possible network configurations. Indeed, inserting a job in some part of the network constrains the possible states of certain switches during job execution. Hence, solving the scheduling problem and the reconfiguration problem at the same time by creating a highly interconnected network of constraints in *temporal*, *topological* (switch states) and *electric* (currents) dimensions, aids finding a solution to both problems.

3. PLANETS

In this section we present the prototype system *PLANETS* ("PLanning Activities on NETworkS"), which will assist the planning engineer at the dispatching service of the Spanish electricity utility ENHER in scheduling the foreseen maintenance activities to be performed in an upcoming week on the power-distribution network, as well re-scheduling them together with unforeseen and urgent repair tasks. The output is a schedule in the form of a Gantt chart completed with a list of switching operations to be able to perform the necessary reconfigurations of the network.

3.1 Overall architecture

The heart of the *PLANETS* system is a constraint engine written in CHIP. It manipulates the temporal, topological and electric domain variables defining the problem, reducing their domains by constraint propagation. The final labeling of temporal and topological variables yields a solution schedule and the associated network reconfigurations. The solution is optimized with respect to the cost function, as will be explained later.

The constraint engine communicates with the outside world by means of a graphical user interface based on XGIP. The interface provides a Gantt chart, clickable topological maps and graphs representing the power-distribution network, and various editors for constraints, jobs, switches and branches.

By means of CLIC (the C-Language Interface to CHIP), a connection is established between the scheduler and ENHER's electric library DISPOT. In this way, load flows and secure-flow patterns can be calculated for evaluation purposes and graphical representation. It also provides an alternative, heuristic algorithm for optimal network reconfiguration based on the one described in [Shirmohammadi 92], and allows manual simulations and manipulations directly on the network graphs. Figure 1 gives a schematic overview.

3.2 Domain Variables

The constraints defining the problem are constraints between three types of (domain) variables: temporal, topological and electric. The *temporal domain variables* represent the starting times of the jobs to be scheduled. Present operations policy at ENHER only allows jobs on the power-distribution network to start and/or end in 15 well-defined time slots in a week. Therefore, the time axis is discretized accordingly and the temporal variables get an initial domain

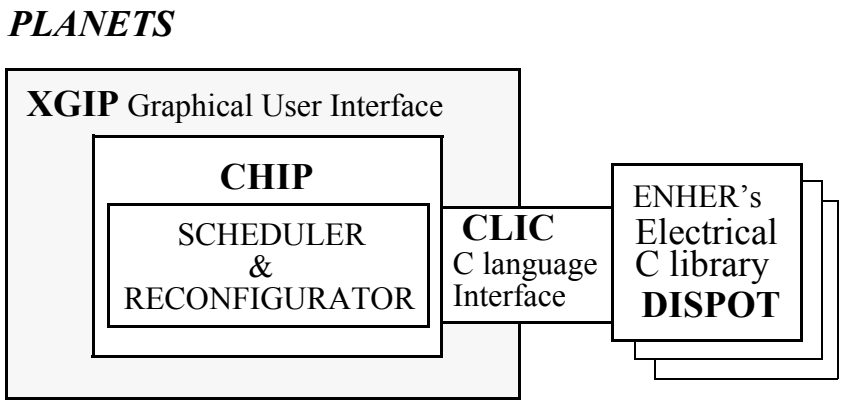

Figure 1. The overall *PLANETS* architecture

ranging from *1* to *15*.

Secondly, the *topological domain variables* represent the states of all switches in each time slot. Hence, with every switch is associated a list of 15 *0-1* domain variables.

Finally, every branch (and switch) of the network needs 15 *electric domain variables* representing the currents that flow in them. Possible current values typically range from *-250 A* to *250 A*. Since we are not interested in specific current values, very few of these electric variables will finally be fully instantiated. Indeed, finding a consistent current labeling does not make much sense: there probably will be hundreds of them and how are we to decide which one will actually take place? The importance of the current variables lies in their lower and upper limits which will propagate changes to the domains of other current variables, as well as to the domains of topological and temporal variables.

We decided not to include domain variables representing voltages in the nodes of the network, because of the additional memory needs and because they do not offer the same propagation effect as current variables due to the very few direct relations with other variables. Voltage levels, useful for evaluation purposes, will be calculated *a posteriori* by means of the interface to the DISPOT library.

3.3 Constraints

Following, we describe the different kinds of constraints used in *PLANETS*:

- **Isolation Constraints:** "*During job execution, the area surrounding the maintained branch must be in outage state*." This is achieved by opening all surrounding switches. The constraint was implemented by means of an extension to the built-in `element/3` constraint, forcing a number of elements in a list of currents (of the maintained branch) or switch states (of the surrounding switches) to be *0*, starting at a position indicated by a temporal domain variable (the start time of the job).

- **Resource Constraints:** "*The available amount of resources must be respected at all times.*" Possible resources required by a maintenance job are vehicles, manpower, etc. The constraint is enforced by a straightforward use of the built-in `cumulative/8`.

- **Precedence Constraints:** "*Jobs on ancestor branches must be carried out before jobs on their descendant branches.*" The constraint translates in a number of inequality relations between temporal domain variables. Apart from these precedences, every job has additionally an absolute priority number. However, these priorities are not considered as "hard" constraints. They merely act as preferences. Violations represent an additional cost per time slot that can be minimized.

- **Consumer Constraints:** "*At all times there must flow a minimal non-zero current to all consumers.*" These constraint will propagate changes to the domains of many other current variables and, doing so, will force the network to reconfigure itself in case of an outage. In case such a reconfiguration would be impossible, we allow exceptionally the isolation of some consumers during a period of at most the duration of the particular job. This exception was necessary to avoid failure of the scheduler and getting a *no* answer. It is implemented by the built-in `atmost/3` constraint: the list of currents supplying any of these consumers can have at most *d* zeros, where *d* is the duration of the particular job. For all other consumers, the constraint is enforced simply by setting the initial domain of their respective lists of current variables.

- **Continuity Constraints:** "*On all nodes, except the root and consumer nodes, the continuity law of Kirchoff must hold at all times.*" This constraint says that the sum of all incoming currents in a node must equal the sum of all outgoing currents. It forms the basis for the propagation of current domains through the network. These constraints are linear equations between domain variables.

- **Switch-behaviour Constraints:** "*An open switch cannot carry current.*" This constraint forms the basis for the topological reconfiguration of the network in all 15 time slots. It is implemented by means of the conditional-propagation construct in CHIP. For all 15 elements of the lists of currents and the lists of switch states, we have:

```
(if S #= 0 then I #= 0),
(if I #\= 0 then S #= 1),
```

- **Radiality Constraints:** "*At all times the network must be radial, i.e., not contain any closed cycles.*" This is enforced by searching all possible cycles and stating that, at any time, at least one of the switches in any cycle must be open, using the `atmost/3` built-in.

- **Overload Constraints:** "*In every branch, current must be below its allowable maximum.*" This constraint translates into simple inequalities on the current domain variables.

- **Energy-demand Constraints:** *"In every consumer branch the domain of the current variable is restricted to a pre-defined profile of energy demands.*

- **Due-date Constraints:** "*Any job must be finished before its due date.*" In the normal case, this is a "hard" constraint. However, the user can indicate that it should be treated as a "soft" one, minimizing violations. Violations again represent a cost per time slot.

The interactions between the above types of constraints and the domain variables on which they act are graphically represented in figure 2.

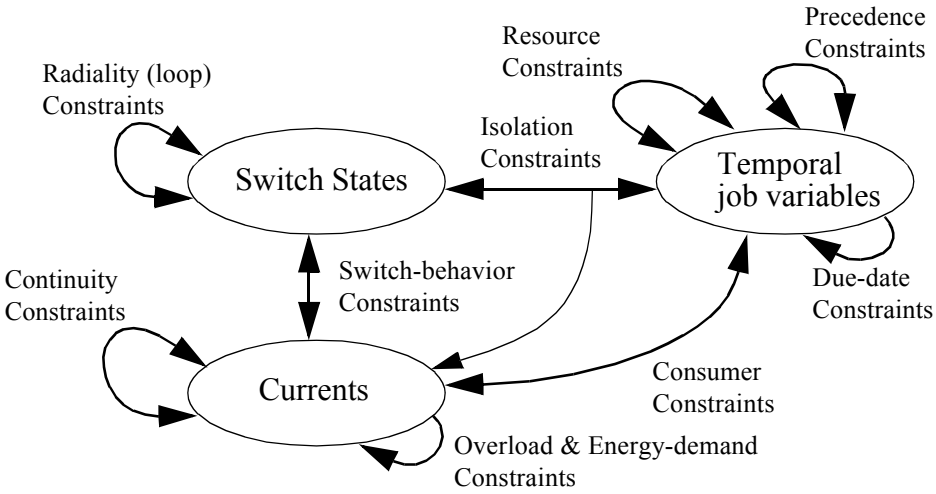

Figure 2. A high-level representation of the constraint network.

3.4 Optimizing the solution

A solution to the overall problem is generated by labeling the temporal and, afterwards, the topological domain variables, yielding a schedule and 15 associated network reconfigurations. Labeling is embedded in a *branch-and-bound* process using the `min_max/2` meta-predicate.

- During temporal-variable labeling, *due-date* and *priority* violations are minimized. The variable-selection heuristic is `most_constrained` and smallest values are tried first.

- During switch-state labeling, a function representing the *global cost of all necessary switching operations* is minimized. For this purpose, every switch has a nominal state and two associated costs: a cost for changing its state (from 0 to 1 or from 1 to 0), and a cost for staying in a non-nominal state during one time slot. These cost terms are added up for all switches and over all 15 time slots, yielding the global cost which is minimized. `most_constrained` switches are labeled first, trying first their nominal state.

4. Implementation and Results

The *PLANETS* scheduler and reconfigurator was completely written in CHIP, making extensive use of the CHIP++ object system and of the module XGIP for the graphical user interface. The CHIP interpreter was linked together with the electric C library. The CLIC module makes the interpreter recognize the new "electric" predicates. The present system runs on Sun workstations and PCs.

For a power-distribution network of about 1200 nodes and 400 operable switches, and 15 maintenance jobs to be scheduled, the system creates about 22000 domain variables. The total time needed to produce the optimal solution with respect to the cost functions at hand, is of the order of 2.5 minutes CPU time on a SuperSparc20. 70% of that time is used to create the variables and set up the constraints; the remaining time is needed for variable labeling and branch-and-bound minimization.

The system has been tested on multiple real-life cases with highly satisfactory results. At the moment it is being transferred to ENHER's dispatching center, where it will go through a second, comparative test phase in parallel with the actual human schedulers. Afterwards, it will be fully interfaced with their Orthogonal Graphics System.

5. Conclusions and future work

In view of the results of *PLANETS*' first prototype, it is clear that the technology of constraint logic programming is well suited for tackling the problem of optimal maintenance-schedule generation on a power-distribution network.

The proposed system improves the exploitation of the distribution network. From an economical point of view, the company obtains an extra benefit by minimizing the total amount of undistributed energy due to forced maintenance outages. Additionally, the global cost of carrying out the maintenance schedule is lowered due to the efficient use of the finite resources and the elimination of many redundant operations.

Finally, the planning engineer can react quickly and with minimum delay to a distribution operator's request for a solution to a problem in the network.

The most important remaining drawback of the present version of *PLANETS* is its memory consumption. The representation of a network of the above mentioned size as domain variables and constraints can easily take up 25 Mb of memory. However, the present representation is too rich, i.e., it contains a lot of redundant information. Presently, we are working on an algorithm to *compress* the information in a distribution network. Working with the resulting compressed network will allow us to reduce considerably the amount of memory needed, while at the same time execution time will be significantly shorter.

6. Acknowledgements

This project was financed by ENHER, S.A. (Empresa Nacional Hidroeléctrica del Ribagorzana). We would like to express our gratitude to Manel Batlle of ORIGIN, S.A. for his valuable help in the development of the interface with the DISPOT load-flow library.

7. Bibliography

[Aggoun 92] Aggoun, A., Beldiceanu, N., Extending CHIP in Order to Solve Complex Scheduling and Placement Problems. *Actes des premières journées francophones sur la programmation en logique*, Lille, France, 1992.

[Aoki 88] K. Aoki, H. Kuwabara, T. Satoh, M. Kanezashi. "An Efficient Algorithm for Load Balancing of Transformers and Feeders by Switch Operation in Large Scale Distribution Systems". *IEEE Trans. on Power Delivery.* Vol. 3, No. 4, pp. 1865-1872, October 1988.

[Filkorn 91] Filkorn, T., Schmid, R., Tidén, E., Warkentin, P., Experiences from a Large Industrial Circuit Design Application, in *Proc. International Logic Programming Symposium*, 581-595, 1991.

[Heintze 91] Heintze, N., Michaylov, S., Stuckey, P., CLP(？) and Some Electrical Engineering Problems, in *Journal of Automated Reasoning*, 9, pp 231-260, 1992.

[Jung 93] K. h. Jung, H. Kim, Y. Ko. "Artificial Neural-Network Based Feeder Reconfiguration for Loss Reduction in Distribution Systems". *IEEE Trans. on Power Delivery*, Vol. 8, No. 3, pp. 1356-1366, July 1993.

[Le Pape 94] Le Pape, C., Constraint-Based Programming for Scheduling: An Historical Perspective, Working Paper, *Operations Research Society Seminar on Constraint Handling Techniques*, London, United Kingdom, 1994.

[Liu 88] C. C. Liu, S. J. Lee, S. S. Venkata. "An Expert System Operational Aid for Restoration and Loss Reduction of Distribution Systems". *IEEE Trans. on Power Systems*, Vol. 3, No. 2, May 1988.

[Nara 92] K. Nara, A. Shiose, M. Kitigawa, T. Ishihara. "Implementation of Genetic Algorithm for Distribution Systems Loss Minimum Reconfiguration". *IEEE Trans. on Power Systems.* Vol. 7, No. 3, pp. 1044-1049, August 1992.

[Nerode 93] Nerode, A., Kohn, W., Hybrid Systems and Constraint Logic Programming, *Proc. 10th International Conference on Logic Programming*, pp 18-24, 1993.

[Puget 94] Puget, J.-F., *A C++ Implementation of CLP*, Technical Report, ILOG, S.A., 1994.

[Shirmohammadi 92] Shirmohammadi, D., "Service Restoration in Distribution Networks via Network Reconfiguration", in *IEEE Trans. on Power Delivery*, Vol.7, No.2, pp. 952-958, April 1992.

[Simonis 87] Simonis, H., Dincbas, M., Using Logic Programming for Fault Diagnosis in Digital Circuits, *German Workshop on Artificial Intelligence (GWAI-87)*, Geseke, Germany, pp 139-148, September 1987.

[Simonis 88] Simonis, H., Nguyen, H.N., Dincbas, M., Verification of digital circuits using CHIP, *Proceedings of the IFIP WG10.2 International Working Conference on the Fusion of Hardware Design and Verification*, Glasgow, Scotland, July 1988.

[Simonis 89] Simonis, H., Test Generation using the Constraint Logic Programming language CHIP, in *Proceedings of the 6th International Conference on Logic Programming*, 1989.

[Simonis 90] Simonis, H., Graf, T., Technology Mapping in CHIP, *Technical Report TR-LP-44*, ECRC, Munich, Germany, 1990.

[Tsai 93] M. S. Tsai, C.C. Liu, V. N. Mesa, R. Hartwell. "IOPADS (Intelligent Operation Planning Aid for Distribution Systems)". *IEEE Trans. on Power Delivery*. Vol. 8, No. 3, pp. 1562-1569, July 1993.

[Van Hentenryck 89] Van Hentenryck, P., *Constraint Satisfaction in Logic Programming*, MIT Press, 1989.

[Wong 87] Wong, K. P., Cheung, H. N, "Artificial Intelligence Approach to Load Allocation in Distribution Substations". IEE Proceedings, Vol. 134, Part C, No. 5, September 1987, pp. 357-365.