PROTOS: Towards Better Production Planning Systems[1]

Jürgen Sauer
Universität Oldenburg, FB Informatik
Postfach 2503, D-26111 Oldenburg

Abstract

Within the EUREKA project PROTOS (Prolog Tools for Building Expert Systems) [Appelrath 87] a system is developed which allows the use of heuristic knowledge in solving specific production planning problems. Problem area is scheduling using the production structure of the chemical industry. Scheduling is a part of production planning with the main objective of creating a schedule of the production which considers all orders (especially the due dates), the availability of resources (e.g. apparatus, raw materials, personell), production requirements, and some other constraints. Usually scheduling is the task of an expert production planner, who has to create the schedule "by hand" using his experience and a lot of data provided by various media (e.g. flow charts, recipes, order lists, production requirements, information about products). Scheduling approaches from Operations Research have their problems with the combinatory complexity of the problem area and lack flexibility in the case of changing situations or parameters. Our aim is to present a system which provides the knowledge and experience of a production planner to advice scheduling and to increase flexibility in planning and replanning. This includes generating a "good" plan (not necessarily an optimal plan) and replan in the case of unforeseen events (e.g. apparatus-breakdowns or lack of resources).

1. The Problem

Problem area of PROTOS is production planning and scheduling within chemical industry. Main task of production planning and scheduling is the creation and adjustment of a schedule which determines the sequence of actions to be taken to realize all given orders. Using a set of input informations the expert planner has to create the schedule of production and to replan i.e. to correct the schedule in case of unforeseen events. Input informations he uses are described below.

- The **main plan** contains a list of orders to be executed for the manufacturing of certain products along with information about amount, start and due dates of the production. Example of one main plan entry: the production of 1000 tons of product P1 shall be started after 4.2.1989 and finished before 15.5.1989.

---

[1] in: Proceedings 14th IFIP Conference "System Modelling and Optimization", Leipzig 1989, Lecture Notes in Control and Information Sciences 143, Springer, 1989.

- The **production requirements** (recipes) are requirements for the manufacturing of certain products comprising among other things the steps to be executed, possible apparatuses, duration of individual steps, cleaning times, jobs and personnel involved. Figure 1 is a graphic representation of production requirements associated with parts of the factory (some of the information mentioned above has been omitted to keep the diagram uncluttered) called flow chart. Apparatuses are indicated by dotted ellipses and rectangles, the links are represented by solid arrows.

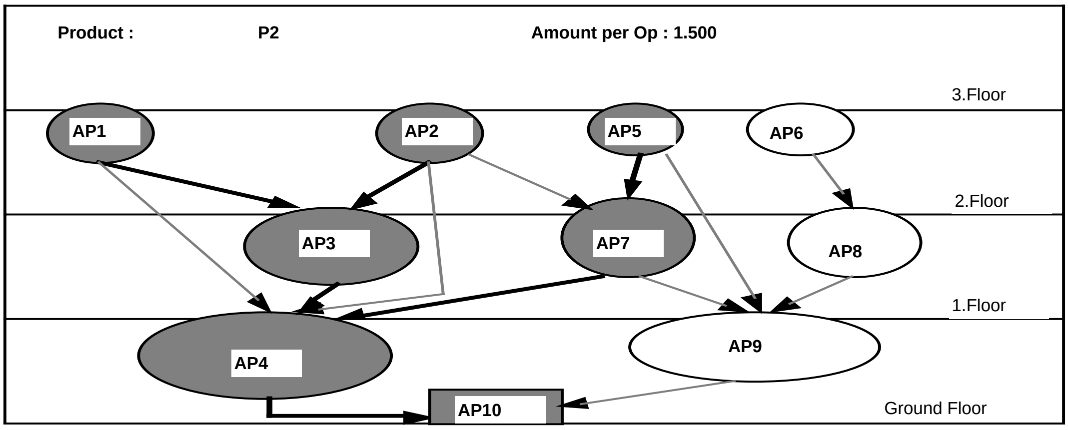

Fig. 1: Flow Chart

- **Product characteristics** are typical product informations such as quantities to be produced, number of steps, number of variants.
- **Descriptions of apparatuses** consist of information about size, type and category of apparatuses.
- **Secondary conditions** are prescriptions to be observed during production, e.g. decrees.
- The **expert knowledge** of production planners consists of knowledge about the appropriate sequences of production processes, what to do in critical situations, etc.

Furthermore the planner has to obtain several requirements and constraints. Among them are
- **production-technical requirements** such as the avoidance of conflicts, the adherence to all of the production requirements, and the observance of resource availability,
- **economic conditions** such as the observance of fixed due dates and production intervals, the minimizing of inventory, and the maximizing of apparatus utilization.

The introduction of an **evaluation function** allows a comparison of different possible solutions (plans) by assigning a (numerical) value to a solution that has been found. This value can then be compared to values of other solutions. It is now possible to obtain an 'optimal' plan. A plan is optimal if it of all the possible plans yields the minimal, or maximal as the case may be, value with respect to the evaluation function. Possible evaluation functions are the total amount of due dates exceeded, the maximal lateness (due date excessions) of an order, or the total production time.

The result (output) of the planning is a **schedule of production** which satisfies (most of) the given requirements and shows the sequence of actions for all given orders. Usually a schedule is presented as a Gantt chart (see figure 2).

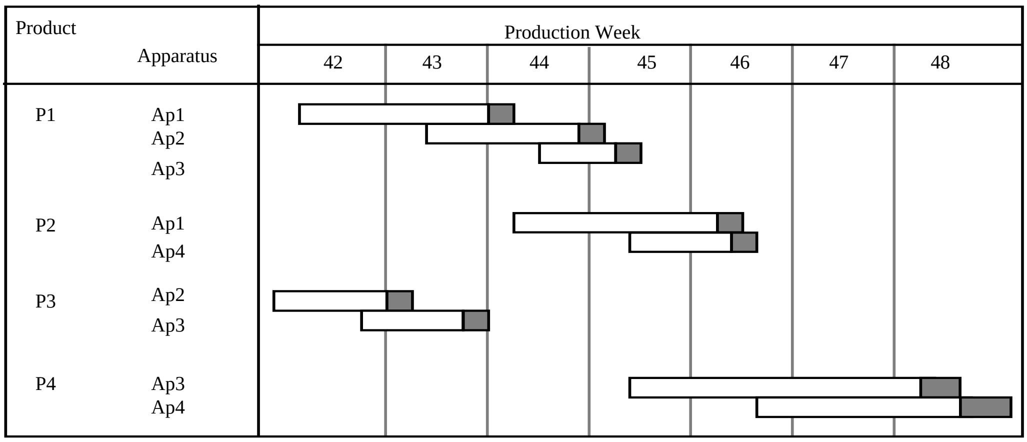

Fig. 2: Production Schedule

The **planning process**, as it is done "by hand" up to now, can schematically be represented by the following procedure. The AND/OR tree of figure 3 gives a visualization of the problem space of this planning process.

- Given a main plan with n orders.
  Until all orders are scheduled do:
  - Select an order to be scheduled.
  - Select an interval for the execution of this order.
  - Select a production variant for the manufacturing of the product.
  - Try to schedule all manufacturing steps of the variant.
  - For every step concerned select a possible apparatus for manufacturing.
  - If there is a conflict, try to resolve it.

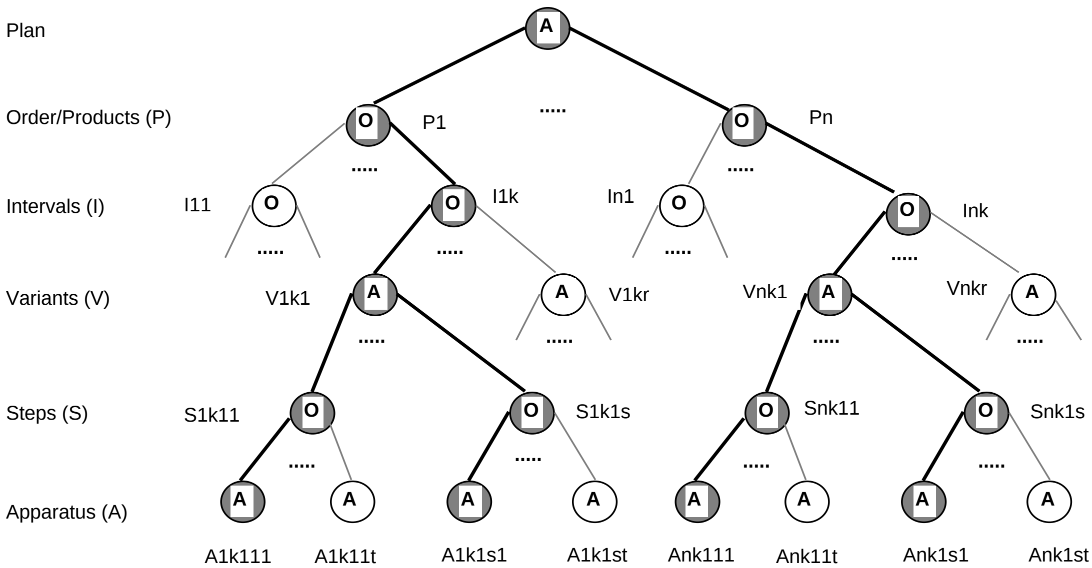

Fig. 3: AND/OR Tree of Planning

The nodes of the tree contain a statement about the contribution of the children nodes to the solution (for an AND-node (A) all children nodes belong to the solution, for an OR-node (O) only one) and, additionally, a value that can be of different domains at each level. Edges denote the links between nodes.

Finding a solution for the scheduling problem is equivalent to finding a solution for the AND/OR tree. In figure 3 a solution is indicated by the solid lines.

An estimation of the size of the AND/OR trees generated can be obtained as follows:

- The plan consists of n orders to be executed, thus n! possible sequences exist for n orders.
- An order can be scheduled in k_n intervals, there are then $k_n!$ interval sequences per order.
- r_{k_n} variants of a product can be tested in one interval, there are thus $r_{k_n}!$ possible sequences of intervals.
- s_{rk_n} steps are associated with each product variant to be scheduled; the steps can be executed in $s_{rk_n}!$ sequences.
- In each step there are t_{srk_n} apparatuses to choose from, here too $t_{srk_n}!$ sequence possibilities exist.

This demonstrates the combinatory size of the problem space of scheduling. If we use heuristic knowledge we can reduce the number of possible sequences drastically. For every level of the tree it is possible to find heuristic rules which will reduce the search space on that level.

2. Knowledge Based Production Planning

The PROTOS approach of a knowledge based system for production planning and scheduling consists of a model of the problem area and an algorithmic approach to create production schedules combined with an user-interface providing all informations and supporting scheduling "by hand". In the following mainly the model and the algorithmic solution will be described.

2.1 The Model

One of the first tasks within the project was the specification of the problem area using a formal modelling language. The aim was to model not only objects and relations (constraints) on the objects (e.g. mathematical models), but also to model the rules and operations which can be used to solve a given planning problem. The modelling is done in PKM (Prolog Knowledge Model) [Ester 89] and CCL (Consistency Constraint Language) [Cronau 88], both developed in another subproject of PROTOS. Chapter 2.1 describes some object classes together with the corresponding attributes of the individual objects. Chapter 2.2 presents an assortment of operations allowed on the objects. In Chapter 2.3 examples are given for differentiated retrieval, inference, consistency and control rules. A detailed model can be found in [Sauer 1988].

2.1.1 Modelling Objects with Attributes and Operations

Objectclasses are described by a name (after CLASS), a list of attributes (ATTRIBUTES) and their types and an identification part. Capital letters denote atomic values (e.g.: CARDINAL, STRING). REQUIRED denotes a required, UNIQUE a unique value and MULTI-VALUED a not atomic, i.e. multivalued area (e.g. a list). The IDENTIFICATION-part contains all attributes neccessary for identifying one instance of the class.

```
CLASS  main_plan                /* contains the information of the given main plan  */
        ATTRIBUTES
             ordernumber           : CARDINAL            UNIQUE, REQUIRED
             productname           : Name_of_product     REQUIRED
             deadline              : Yearmonth           REQUIRED
             amount                : CARDINAL            REQUIRED
             orderpriority     : CARDINAL          REQUIRED
        IDENTIFICATION
             ordernumber.

CLASS  production_requirements  /* contains all informations about the production process */
        ATTRIBUTES
             productname           : Name_of_product     REQUIRED
             production_step       : CARDINAL            REQUIRED
             apparatus             : Apparatus_name      MULTI-VALUED
             . . .
        IDENTIFICATION
             productname, production_step.
```

Classes are also used as type declarations, e.g.:

```
CLASS   Apparatus_kind
        ATTRIBUTES        kind     : (email,stainless,rubber-inlet,wood).
```

Other important classes of objects are:

product_description	necessary intermediates and raw materials for products
production_plan	all successfully planned production orders
stock_list	the actual stock of raw materials, intermediates, and products
apparatus_list	the actual state of the apparatuses.

The allowed operations on object classes are INSERT, DELETE and MODIFY. Upper case letters are denoting variables, which have to be instantiated before calling the operation.

```
DELETE   production_plan                     /* Delete one entry of production_plan  */
              ordernumber          : N.
              production_step      : F.

MODIFY   main_plan                   /* Change an entry in the main_plan */
              ordernumber    : O
              productname          : Pr
              deadline             : D
              amount         : M
              orderpriority  : P.
```

2.1.2 Modelling Rules

Rules are of main importance in representing the planning knowledge of the production planner. The heuristic planning knowledge can be represented by rules, this will be done by using inference- and control-rules (for example inference-rules represent knowledge how to create new production orders if there are overlappings, and control-rules represent knowledge how to use the inference-rules the ’right way’. Two additional kinds of rules will be described, retrieval- and consistency-rules which are necessary in modelling the planning area. The question of classifying rules and what rule belongs to which class of rules is still under discussion.

a) Retrieval-rules

Retrieval-rules in PKM are defined as Retrieval-Operations. Upper case letters denote variables, which have to be instantiated before calling the rule; ... means, that the RETRIEVE NEXT operation has to be executed again and again until all information wanted is found (it is user-driven, how often RETRIEVE NEXT has to be executed). Examples are :

```
/* Retrieve all informations of a product of the mainplan */
        DEFINE RETRIEVAL
                INSTANCE      main_plan
                ATTRIBUTES  ALL
                WHERE  productname = P.
        RETRIEVE NEXT.
        ...
        FINISH RETRIEVAL.
/* Retrieve all informations of one production_step N of product P */
        DEFINE RETRIEVAL
                INSTANCE  production_requirements
                ATTRIBUTES  ALL
                WHERE  productname = P AND production_step = N.
        RETRIEVE NEXT .
        FINISH RETRIEVAL.
```

b) Consistency-Rules

Consistency-rules are used to provide a consistent knowledge /data base. They are described in CCL created together with PKM. Every consistency-rule definition consists of the name (after CONSTRAINT CLASS), a variable declaration part (LOCALS or PARAMETER) and the definition of the consistency constraint (ATTRIBUTES). Variables start with an uppercase letter.

Rule C-1. The production_plan contains only available apparatus.

```
CONSTRAINT CLASS Rule_C_1
        LOCALS  Fplan : production_plan, Blist : apparatus_list
        ATTRIBUTES
            constraint:     FOR EVERY Fplan IN production_plan :
                            FOR SOME Blist IN apparatus_list :
                                apparatusname OF Fplan = apparatus of Blist
                                AND   available OF Blist  = TRUE.
```

Rule C-2. The production_plan of the next 14 days contains only products, where all intermediates and raw materials are available.

```
CONSTRAINT CLASS Rule_C_1
        LOCALS  Fplan : production_plan, Pdesc : product_description, Stock : stock_list
        ATTRIBUTES
            constraint    :      FOR EVERY Fplan IN production_plan :
                                 IF  starttime OF Fplan  ≤ (day OF today + 14)
                          THEN
                                     FOR SOME Pdesc in product_description:
                                         productname OF Pdesc = productname OF Fplan
                                         AND
                                         amount OF Fplan ≤ stock OF stock_list(Stock)
                                             WHERE productname of Pdesc = name OF Stock
                                 ENDIF.
```

Other possible consistency-rules are :

C-3. The value of the attribute 'productionsteps' in 'product' must be equal to the number of production steps of the corresponding product in 'production_requirement', there must be an entry for all steps from 1 to N.

C-6. After the "planning" for every production_order there has to be a corresponding entry in the production_plan or in the list of overlappings.

c) Inference-rules

Inference-rules are used to infer 'new' facts from the given facts and are described using the syntactical prescription INFERENCE RULE. Every inference-rule has a name (after INFERENCE RULE), a list of variables (FOR ALL), a condition part (IF), and a conclusion part (THEN).

Rule I-1. If there is no overlap, then an order can be planned on the desired deadline.

```
INFERENCE RULE rule_I_1
        FOR ALL       Px, An, StartDay, EndDay, ANr, M
        IF            production_order
                           ordernumber          : ANr
                           production_step : M
                           productname          : Px
                           apparatusname : An
                           starttime            : StartDay
                           endtime         : EndDay
              AND NOT rule_I_3(ANr, Px, An, StartDay, EndDay)
              AND    INSERT
                      production_plan
                           ordernumber          : ANr
                           production_step : M
                           productname          : Px
                           apparatusname : An
                           starttime            : StartDay
                           endtime         : EndDay
        THEN          rule_I_1(ANr).
```

Rule I-3 : There is an overlap, if a planned order and an order to be planned use the same apparatus on the same day.

```
INFERENCE RULE rule_I_3
        FOR ALL     Px, An, Py, Start, End, StartPlan, EndPlan, ANr, ANrPlan, M
        IF          production_order
                         ordernumber          : ANr
                         . . .
             AND    production_plan
                         ordernumber          : ANrPlan
                         . . .
             AND    EndPlan > Start
             AND    StartPlan < End
             AND   INSERT
                    overlap
                         ordernumber          : ANr
                         . . .
        THEN       rule_I_3(ANr, Px, An, Start, End).
```

Examples of other inference-rules are :

- I-2. Orders with higher priority (maximum priority = 9) have to be planned first.
- I-5. If we look for an alternative apparatus of a stainless-apparatus, we can use an other stainless-apparatus or an email-apparatus of the same size, if it is available.
- I-7. If an order has to be changed because of an overlap, create a new deadline.
- I-8. If an order has to be changed because of an overlap, reduce the amount to produce.
- I-9. If an order has to be changed because of an overlap, look for an alternative apparatus.

d) Control-rules

Control-rules are used to express how to manage the application of the other rules. In PKM it is not possible to describe control-rules explicitly. PKM is based on the control-strategy of PROLOG, so the sequence of inference-rules may be used to control the use of rules. To provide an explicit representation of control-rules an extension of PKM is under discussion. Some of the possible control-rules are :

- K-1. Order main-plan by priority.
- K-2. Start planning at the end of the planning interval.
- K-4. If there is an overlap, try rules I-7, I-9, I-8.
- K-7. Rule K-1 has to be applied before rule K-2.
- K-8. If the overlap is caused only by one or two steps of the production, then try rule I-9.
- K-9. If there is an overlap, apply rule K-8 before rule K-4.

2.2 The Planning System

The proposed approach of a knowledge based production planning system bases on the idea of having a two layer planning structure consisting of a Basic- and a Control-Algorithm. The Basic Algorithm is used to efficiently create a production schedule by using only few heuristic informations and the control strategy of Prolog. The Control-Algorithm provides more flexibility in the use of heuristic information both for creating and adjusting production schedules. The system architecture is shown in figure 4.

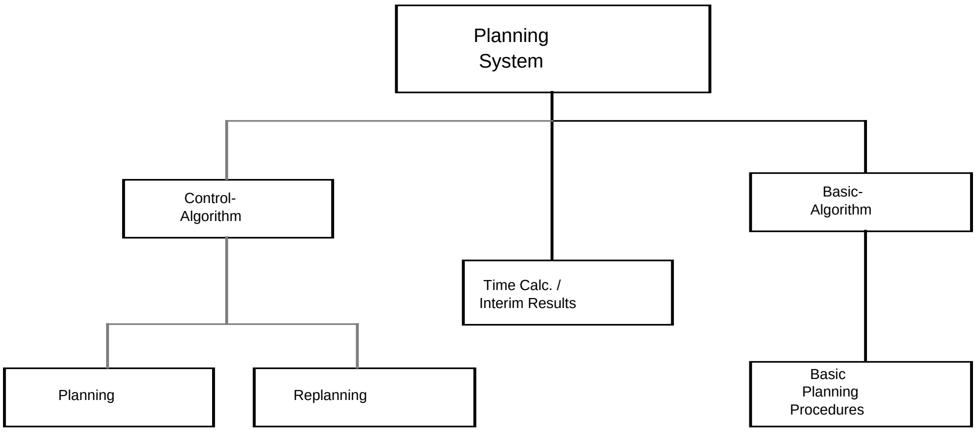

Figure 4. System architecture

2.2.1 The Basic-Algorithm

The basic-algorithm models the planning strategy of the production planner shown in chapter 1 in the following scheme. The heuristics used in every 'choose'-part are mentioned after 'using'.

1. check preconditions
2. choose order using
   - plan products with less production alternatives first
   - plan products that have to be finished earlier first (FIFO)
   - plan products with smallest planning interval first
   - user given priority,

   if all orders planned -> FINISH,
3. choose production interval using
   - try the given start-date first, go back one week, and then go in future direction,

   if order planned -> 2.
4. choose variant using
   - try stem variant 0 first, then variants with increasing number,

   if order planned or not plannable -> 3.
5. choose step using
   - try step with minimum of alternative apparatus first,

   if all steps planned or one step not plannable -> 4.
6. choose apparatus using
   - try main apparatus first, then alternatives,

   if apparatus free or no alternative left -> 5, else -> 6.

The Basic-Algorithm is implemented in Prolog (BIM_Prolog, Quintus-Prolog) using the problem solving strategy of Prolog (depth first search with backtracking). Integrated in an user-interface developed by an other partner of PROTOS it was presented on the CeBIT 89 and on WorldTech Vienna. Some data involving quantity dimensions and first runtime measurements are summarized in the following table.

Parameter / Building	939	31	
products	33	54	
Ø steps per product	10	3	
Ø alternative-app. per step	4	3	
Ø variants per product	2	1	
apparatus	54	32	
orders	44	50	
planning-period (month)	6	6	
runtime Basic-Algorithm (BIM-Prolog) (sec)	165	23	

2.2.2 The Control-Algorithm

The Control-Algorithm is under development and shall provide more heuristics in planning to increase flexibility and plan quality, and it shall provide heuristics for replanning in the case of unforeseen events. Some of the main characteristics are

- the integration of the Basic-Algorithm,
- the selection of methodologies and heuristics for planning, so to build an "own" planning algorithm for a given situation
- the interactive selection of heuristics usable for replanning in specific situations,
- the integration of "new" heuristics by the user.

First experiences are made using different planning methodologies combined with different heuristics at the 'choose'-parts of the methodologies.

References

[Appelrath 87] Appelrath, H.-J.: "Das EUREKA-Projekt PROTOS", in: Proceedings "2. Int. GI-Kongress '87 Wissensbasierte Systeme", IFB 155, Springer, Heidelberg, 1987.

[Cronau 88] Cronau, J.: "CCL : A Language for Defining Consistency Constraints", in: [PROTOS 88].

[Ester 89] Ester, M.: "Konsistenzwerkzeuge für Prolog-Wissensbasen", Dissertation, ETH Zürich, 1989.

[PROTOS 88] Appelrath, H.-J., Cremers, A.B., Schiltknecht, H.: "PROTOS First Workshop", Sandoz AG, Basel, 1988.

[Sauer 88] Sauer, J. : "Model of the PROTOS-Production-Planning-Problem", in: [PROTOS 88].