Computer-Aided Planning of Purge Operations

A strategy is proposed for synthesizing operating procedures for purging species from chemical processing systems during start-up. This strategy is based in part on the artificial intelligence methods of planning with constraints and symbolic functional modeling of chemical processing systems. Techniques are introduced for identifying chemical species to be purged, choosing methods and auxiliary fluids for executing purge tasks, and allocating chemical species to their final destinations. The implementation of this strategy is illustrated with the example of purging a hydrocarbon chlorination system during start-up.

R. H. Fusillo, G. J. Powers
Department of Chemical Engineering
Carnegie-Mellon University
Pittsburgh, PA 15213

Introduction

Purge operations remove unwanted materials from a chemical processing system. A system may need to be purged during start-up to prevent potentially dangerous or corrosive mixtures, or before vessel entry to remove toxic or flammable materials. Planning purge operations entails discovery of undesirable components in the system, devising methods for removing these components, and sequencing the resulting tasks. Better planning of purge operations may improve the integrity of the instructions by which operating technicians perform their jobs. Product quality might be enhanced by reducing the risk of contamination or corrosion. Carefully planned procedures would also help to avoid toxic releases from the plant.

This paper presents a method for computer-aided synthesis of purge operations. Issues addressed include constraint-guided identification of components to be purged, rule-based choice of methods and auxiliary fluids for purging, species allocation, and a search technique for feasible sequencing of purge tasks. The use of constraints for determining feasibility and reducing search effort is emphasized.

Prior work in operating procedure synthesis

Early approaches to operating procedure synthesis were based on synthesis of valve sequences subject to simple operating constraints (Rivas and Rudd, 1974; O'Shima, 1978) and generation of start-up procedures through more general state transitions (Kinoshita et al., 1981; Ivanov et al., 1980a,b). The valve sequencing methods were effective, but their representation of the system as a network of valves and connectors limited the problems that could be solved to establishing flow between two points in the network. The latter class of methods was general enough to allow higher-level models, but left certain procedural and representational details underdeveloped. The paper by Kinoshita et al. introduced the important notion of system decomposition for procedure planning. Current researchers (Fusillo and Powers, 1987; Stephanopoulos, 1987) are applying modern artificial intelligence technology to operating procedure synthesis.

In previous work (Fusillo and Powers, 1987) we devised an approach to operating procedure synthesis based on a state-space search for plans. This approach combined means-ends analysis with constraint-guided planning and a symbolic modeling technique. An experimental program called POPS (Prototype Operating Procedure Synthesis Program) was introduced, in which some of our methods were implemented. In this paper, we illustrate the extension of this methodology to the purging of chemical processing systems.

In planning purge operations, it is important to individually consider each species to be purged as well as interactions that occur between them. For each species to be purged, the following must be selected:

1. A purgative—a fluid that pushes out and/or dilutes the purged component
2. A method—the manner in which the purge operation is accomplished (e.g., a system can be swept through with a gas, evacuated using a pump, etc.). The choice of method is based, in part, on the phase of the material to be purged; Figure 1 shows possible purge methods organized by phase.
3. The destination—the location to which the purged material is to be removed

When several components are to be purged, multiple possibili-

Correspondence concerning this paper should be addressed to G. J. Powers.

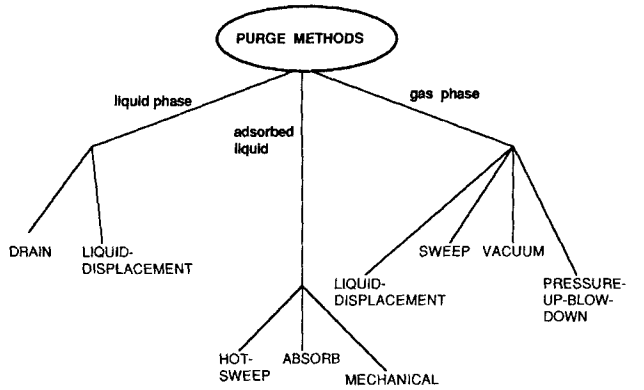

Figure 1. Organization of purge methods by phase of forbidden component.

ties of purgatives, destinations, and methods exist for each component. Further, there are multiple orderings that are possible for every selection. Hence, for most purging problems there is a large number of possible sequences of procedures.

The next section of this paper includes a discussion of a new theory of purge operations planning. Symbolic modeling techniques and a new method for goal formulation are discussed. The formulation and use of constraints is illustrated for efficiently determining a feasible plan, operator selection, and ordering of operators. An example problem section covers application of these techniques to the problem of purging a reactor system.

Theory and Implementation

In the present theory of operating procedure synthesis, the operation of a chemical processing system is viewed as a sequence of goal states. States are defined by vectors of physical quantities, such as temperature, pressure, composition, and others defined at appropriate locations within the process. The synthesis of operating procedures is the problem of finding a set of actions, or operators, that brings the system from some initial state to the desired goal state. In this state-space view, an operator represents a manipulation of the process equipment.

Applying the concept of hierarchical planning (Sacerdoti, 1977) to operating procedure synthesis, planning proceeds at progressively finer levels of detail, from operations involving entire subsections down to individual valve or set point manipulations. Purge operations, for the purposes of this work, are defined at a relatively coarse level of detail. A single purge operator (in the state-space sense) represents the action of purging at least one phase contained in a subsystem. Choices of method, purgative, and destination (M, P, D) are specified for a purge operator.

In this section, a theory of purge operation synthesis is presented. A planning method is given that involves

1. Identifying the conditions that must be changed and compiling a list of "forbidden components" (goal formulation)
2. Translating the changes into operating tasks (operator generation)
3. Ordering the operators so that the overall goals of plant operations are satisfied (task sequencing)

The application of a symbolic modeling technique will be discussed, as will the formulation and utilization of symbolic constraints.

Constraints

In his research on planning, Stefik (1981) identified three major roles of constraints in a planning system:

1. Reducing the search space by pruning planning paths as early as possible
2. Acting as a communications medium between interacting subproblems
3. Serving as partial descriptions of objects or sequences to be chosen

It is the last of these roles that is the basis for a decision strategy called least commitment. Constraints are used to rule out infeasible selections as early as possible, but options are kept open as long as possible. The least commitment strategy is used in the present work to enable the planner to find all feasible sequences of purge operations. Constraints are used in all three stages of planning: goal formulation, operator generation, and task sequencing.

The operation of chemical plants is normally a high constrained domain. Many constraints are generated *a priori* by considering preconditions for unit operations, requirements for a reaction, production requirements, environmental and explosion hazards, and materials of construction (Fusillo and Powers, 1987). We make the distinction between local and global constraints because of the source of the constraints in the processing network and the way they are used in the planning process. Local constraints are due to preconditions for process tasks, for instance:

DO NOT OPERATE HEATER HTR-25 WITH NO FLOW IN ITS TUBES (to avoid damage to tubes).

An example of a global constraint due to system chemistry is:

DO NOT MIX Cl_2 *AND* CH_4 (to avoid explosion hazard).

In the current version of POPS, all global constraints must be entered by the user. Most local constraints are generated automatically from detailed equipment models. Special-purpose local constraints can be added by the user. All constraints in POPS are expressed symbolically, as Lisp predicates.

Planning

Planning refers to devising a course of actions to achieve one or more goals. Here, the approach introduced by Fusillo and Powers (1987) is extended to handle the complexities of planning purge operations. The present emphasis of this work is on generating feasible sequences of operating procedures. Future work in this area will include improved modeling methods and the development of evaluation functions that will allow the selection of optimal sequences.

Goal Formulation. We previously posed planning for procedure synthesis as a means-ends analysis problem where the initial and goal states are characterized by vectors describing the process at both ends of planning. In the domain of purge planning, however, goals sometimes cannot be discovered by difference detection, as is characteristic of means-ends analysis. Normally, it is not known at the start of planning whether chemical components present at an initial state are allowable at the goal state. In cases of this nature, purge goals are discovered by analysis of global constraints. A hypothetical state vector is formed consisting of the goal state vector augmented with the chemical components present at the initial state. Global and local constraints for the system are then evaluated to determine whether the hypothetical state vector is feasible. Each constraint that is violated is analyzed, and if one of the augmented components

(those initially present) is causing (or involved in) the infeasibility, it is added to a forbidden-components list. Following this analysis, a goal is formulated to purge each member of the forbidden-components list.

Our method of goal discovery can produce sets of goals that are not independent. While this may lead to some duplicated effort in searching for operators and other elements, it will not cause planning to fail. The task-sequencing procedure of POPS is able to recognize that multiple goals are satisfied by one operator, thus eliminating the redundancy arising from dependent goals.

Operator Generation. Operators in the domain of purge planning represent subsystem-wide purge operations, with selections for method, purgative, and destination specified. Planning of purge operations proceeds hierarchically. The first manipulations to be proposed are macro-level; that is, the direction "PURGE O_2 by *SWEEP* WITH N_2 TO DESTINATION *vent-2*" is proposed as a single operation. The actual purge operation will involve closing all outlet valves except to the desired destination, establishing flow of the purgative, and perhaps establishing bulk flow in the system using a pump or compressor.

The steps of the process for proposing purge operators are:

1. Considering each element of the forbidden-components list separately, an abstract operator is proposed in which selections are not yet made for method, purgative, and destination.

2. For each element of the forbidden-components list, try to find candidates for method, purgative, and destination. This is accomplished by testing a set of rules (productions) for finding these candidates.

3. For each element of the forbidden-components list, combine the candidates for method, purgative, and destination into feasible combinations. This is accomplished by proposing all possible *n*-tuples, or sets of one selection for each of M, P, D. Once proposed, each *n*-tuple is checked for consistency with respect to a set of constraint functions, to be described in this section. Each consistent *n*-tuple represents a fully instantiated purge operator.

At the conclusion of this process, a set of possible ways to purge each individual forbidden component has been generated. These operators are next combined and sequenced to satisfy operating goals and constraints.

An example of an abstract operator is shown below:

⟨Purge op2⟩
Is a: purge-operation
Component: O_2
Phase: gas
Initial value: medium
Final value: 0
Method: ?method-2
Purgative: ?purgative-2
Destination: ?dest-2

The slots for method, purgative, and destination contain symbols beginning with "?" These are unbound variables, which eventually become bound to lists of candidates.

Examples of rules that are tested to find candidate bindings are:

Purge-method-select rule 4:
IF: Phase to be purged is gas and the flow path is direct (i.e., there are not too many dead-ended branches)
THEN: Propose method = sweep with weight = 5

purgative-select rule 1:
IF: Phase to be purged is gas
THEN: Propose all system gases (gases present in system at goal state) as purgatives whose concentrations are above a minimum with weight = 10

These rules find candidates whose feasibility is determined when combined with other candidates (in *n*-tuples). The weights listed with these rules are used heuristically to express degrees of preferences. A listing of M, P, D rules is given in the appendix. The qualifications for applications of methods and purgatives can be expanded to match particular domains, for example, toxic gases, systems with particulates, and others.

Once formed, lists of candidate values for method, purgative, and destination are stored at the variables ?method-2, ?purgative-2, and ?dest-2. *N*-tuples are formed for each forbidden component by creating all possible combinations of M, P, D, and subjecting each combination to the following consistency constraints:

1. The combination of forbidden and purgative components must not violate global constraints under conditions projected to occur during the purge operation (as specified by the "conditions-during" list in the frame for the method specified in the *n*-tuple)

2. The purgative must not be the same as the forbidden component

3. Neither the purgative nor the purged component can violate local constraints at the destination specified in the *n*-tuple

4. The phase of the purgative must be consistent with the "required phase" specified by the method

Task Sequencing. Operators are ordered into sequences that satisfy the operating goals, are feasible with respect to operating constraints, and are nonredundant (i.e., tasks should not be repeated needlessly). In the present work, depth-first tree search is used for operator ordering. A task sequence (tree branch) starts at an initial state and is extended by adding feasible operators until the operating goals are satisfied. If no feasible extension can be found for an unfinished branch, backtracking occurs by the removal of operators from the branch, until a feasible extension can be found.

When *n*-tuples have been created and checked for consistency, the search for feasible orderings of purge operations begins. To accomplish this, one fully instantiated purge operator (formed from an *n*-tuple) is selected for each forbidden component. Given a minimal set of purge operators, a search is conducted for an ordering (or orderings) that is feasible with respect to local and global constraints. This is accomplished by a "double backtracking" algorithm, so-called because backtracking will occur to find all minimal sets of operators and to discover all feasible orderings of each minimal set. The double backtracking algorithm is shown schematically in Figure 2. At the left of the diagram is a set of knowledge sources for the forbidden components. These knowledge sources are the abstract operator frames for the forbidden components, each one augmented by a set of feasible *n*-tuples. The set of *n*-tuples is sorted according to combined weight, which allows the preferred *n*-tuples to be used first. The outer backtracking algorithm is capable of forming all combinations of one operator (*n*-tuple) from each knowledge source. Each set is used by the inner back-

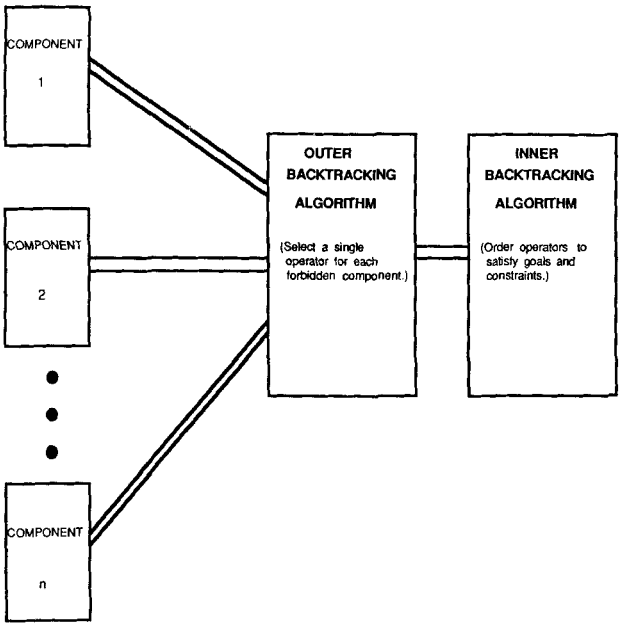

Figure 2. Diagram of double-backtracking algorithm.

tracking algorithm, which can find all feasible orderings of operators. The inner backtracking algorithm performs a depth-first search to order the operators subject to constraints. This depth-first search routine can be stated as follows:

1. Given a search path (ordered sequence of operators), choose an operator that satisfies the following:
   - The operator is not yet on the planning path
   - The operator has not yet been tried at this point on the path
   - The goal for which the operator was proposed is unsatisfied

   If no such operator can be found, backtrack to a prior choice point.
2. Evaluate local constraints for applying this operator at the state existing at the end of the planning path. If the operator is infeasible, go back to step 1.
3. Simulate the state of the system during the operation. These effects are contained in the "conditions-during" attribute of the purge method indicated by the operator. A hypothetical state vector is formed by making the changes to the state vector at the end of the current planning path.
4. Evaluate global constraints at the hypothetical state vector formed above. If an infeasibility is detected, go back to step 1.
5. Simulate the state of the system after the operation. These effects are contained in the "conditions-after" attribute of the purge method indicated by the operator. A hypothetical state vector is formed by making the changes to the state vector at the end of the current planning path.
6. Evaluate global constraints at the hypothetical state vector. If the operator is infeasible, go back to step 1.
7. Place the new feasible operator on the end of the planning path, along with the updated state vector. Mark all goals which have been satisfied by applying the new operator.
8. If all goals are satisfied, exit. Else, go to step 1.

By marking goals as satisfied (and unmarking them if backtracking occurs), the ordering algorithm can take advantage of "multiplexing." That is, one operator may satisfy several goals. Hence, if a purge action removes all forbidden components from one or more phases, then actions need not be planned to remove components that have already been purged.

Modeling

Steps 3 and 5 of the task-sequencing algorithm call for estimation of the state of the system during or following purge operations. In this work, a modeling technique called functional modeling, introduced by Fusillo and Powers (1987) is used. In this modeling scheme, variables represent subsystem-wide levels of process quantities (such as temperature, pressure, flow rates, and concentrations), and take on discrete values (i.e., zero, low-low, low, med-low, medium, med-high, high, high-high).

Knowledge about each type of purge method is represented by a frame which contains simulation information in the form of an add list. Such a frame for purge method type *sweep* is shown in slot-and-filler notation below:

```
Sweep
  Is a: purge-method
  Required phase of purgative: gas
  Phases affected: gas
  Conditions-during   (bulk-flow medium)
                      (<destination open)
                      (<purgative-concentration high)
  Conditions-after:   (components <purgative)
```

This frame contains the information that the purgative fluid must be a gas (this is used to ensure that a compatible purgative will be selected for the method) and assumes that only the gas phase will be affected by the operation. This model is a simplification—in the physical situation, some volatile liquids might be removed. The slot "conditions-during" is used to estimate conditions during the sweep operation. In this case, temperature and pressure are assumed to be the same as prior to the operation. The value of bulk flow is set to "medium," and the composition of the system is taken to be the same as before the purge operation, with the addition of the purgative gas at a concentration of "high." The destination location (usually a vent) is taken to be "open." This last piece of information is used in evaluating local constraints at the destination. The "conditions-after" slot is used similarly, to estimate conditions after the operation. In this case, the system is assumed to be full of the purgative gas after the sweep is performed.

An Example Problem

Consider the simplified chloroform reaction subsystem (Austin, 1984) shown in Figure 3. Chloroform is produced according to the reaction:

$$CH_4 + 3\ Cl_2 \rightarrow CHCl_3 + 3\ HCl$$

This reaction takes place in the gas phase, with chlorine as the limiting reagent. The design of the process is such that chlorine can create explosion hazards if allowed to accumulate in the reaction subsystem; thus there are strong constraints intended to enforce the stoichiometric balance and ensure that the chlorine is completely reacted in each pass through the reactor. The system is equipped with vents to the atmosphere and to a hydrocar-

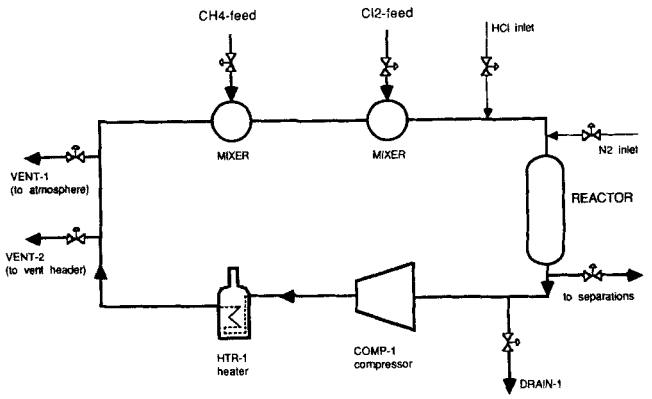

Figure 3. Simplified chlorination flowsheet.

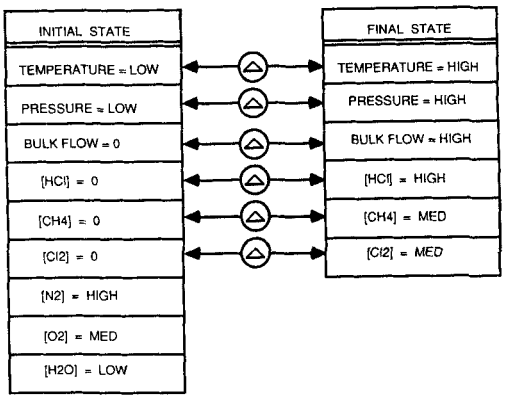

Figure 4. Initial and final state vectors for planning start-up.

bons vent header, a low-point drain for eliminating liquids, and a dry nitrogen feed line.

The relevant global and local constraints for purging this system are presented in Table 1. There are global constraints intended to avoid corrosion and explosion hazards in the system, and local constraints at the process outlets. The local constraints are for avoiding toxic discharges to the atmosphere, explosion hazards in the vent header, and interference with a condenser in the downstream separation system.

Initially, the system is shut down, cold, and open to the atmosphere. Nitrogen and oxygen gases are present in the system, as well as water in the form of vapor and liquid adsorbed onto the walls of the process equipment (termed here an adsorbed liquid). Figure 4 shows the means-ends analysis step of comparing the initial and final states. The encircled deltas in the figure indicate differences in state variables that are detected between the states. Goals are formulated to eliminate these differences. O_2, N_2, and H_2O are not included in the final state vector, as they are not required at the goal state. It must be determined, however, if they are allowable at the goal state. This is accomplished by constraint analysis and interaction with the human process planner.

Table 1. Constraints used in Example Purge Problem

GLOBAL CONSTRAINTS
gl-constr-1
Do not mix chlorine and methane unless the system temperature is high
gl-constr-2
Do not allow chlorine to exceed a stoichiometric amount (for trichlorination) relative to methane
(gl-contr-1 and gl-constr-2 are meant to avoid potentially dangerous mixtures of methane and chlorine)
gl-constr-3
Do not mix H_2O and HCl
gl-constr-4
Do not allow O_2 in the system if the temperature is $\geq$ *medium*
(gl-constr-3 and gl-constr-4 are meant to avoid corrosion)
gl-contr-5
Do not mix methane and O_2
(to avoid an explosion hazard)
LOCAL CONSTRAINTS
lcl-constr-1
Do not open vent-1 if toxic materials (e.g., Cl_2, CH_4, HCl, CH_mCl_n) are present in the system
lcl-constr-2
Do not open drain-1 if toxic materials (e.g., Cl_2, CH_4, HCl, CH_mCl_n) are present in the system
(these two vents open to the atmosphere; the constraints are meant to avoid toxic releases)
lcl-constr-3
Do not open vent-2 if O_2 is present in the system
lcl-constr-4
Do not open vent-2 if chlorine is present in the system
(lcl-constr-3 and lcl-constr-4 are meant to avoid explosion hazard in the hydrocarbons vent header)
lcl-constr-5
Do not open the line to the separator if noncondensible inerts are present (e.g., O_2, N_2)
(propagated from separation system; noncondensibles will interfere with condensation)

The hypothetical state vector shown in Figure 5 is created by augmenting the "final" state vector with the list of components *from the "initial" state vector. The constraints*, Table 1, are evaluated at this state. As shown in Figure 5, the presence of H_2O with HCl at the final state violates the global constraint gl-constr-3 (Do not mix HCl and H_2O). When this constraint violation is detected and one of the unallocated components (H_2O) is found to be involved in the constraint violation (by checking the argument list of the constraint), H_2O is placed on the forbidden-components list. In a similar manner, O_2 is discovered as a forbidden component because of constraint gl-constr-5 (explosion hazard). Discovering that N_2 must be removed requires somewhat more complicated reasoning. Local constraint lcl-constr-5 is propagated from the downstream purifica-

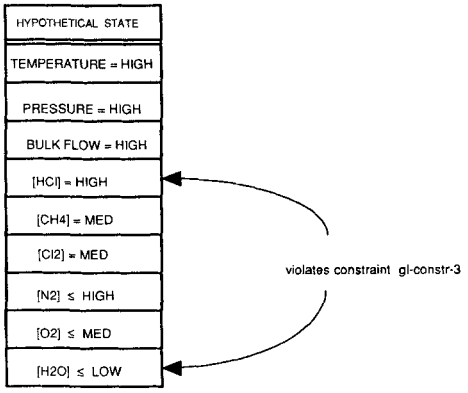

Figure 5. Detection of forbidden component by constraint analysis.

tion system, where inert noncondensible gases reduce the effectiveness of a condenser. This constraint, combined with the knowledge that the connecting stream between the reaction and separation subsystems will be open, forces N_2 to be purged. This last reasoning task has not yet been automated.

Once the forbidden-components list has been compiled, a goal is established to purge each of its members. For each purge goal, an abstract purge operator is established. Operator generation proceeds as discussed previously. Candidates for methods, purgatives, and destinations are found with the aid of sets of rules. Each rule that fires proposes one or more candidate purgatives or methods for the purge task, along with weights which are used to express preference. The process of operator generation is illustrated in Figure 6.

Fully instantiated operators are formed by making definite choices for method, purgative, and destination for purging each component. From the choices shown in Figure 6, there are 30 possible operators for purging O_2 alone. Using a least commitment strategy, constraints are used to prune operators as soon as possible and other choices are delayed as long as possible. Each possible set, or n-tuple, of {M, P, D} is assembled, along with the sum of weights. After the n-tuples are assembled, each one is subject to the consistency constraints listed earlier. The following n-tuple was assembled for purging O_2 from among the possibilities in Figure 6:

(15 sweep HCl vent-1)

which translates to, "Purge O_2 by sweeping with HCl to destination vent-1 (total weight = 15)." The first consistency constraint dictates that the combination of forbidden and purgative components must not violate global constraints under conditions projected during the purge operation, as specified by the conditions-during list for the method. The frame for method *sweep* is given in the theory section. The projected conditions are given by the state vector shown below:

Bulk-flow = medium
Temperature = low
Pressure = low
[O_2] = medium
[HCl] = medium

No global constraints are violated at this state. The second consistency constraint is satisfied, as the purged component (O_2) is not the same as the purgative (HCl). The third consistency constraint dictates that neither the purged component nor the purgative can violate local constraints at the destination. This n-tuple does not satisfy this consistency constraint, as local constraint lcl-constr-1 is violated (HCl cannot be discharged to the atmosphere.) The pruning process is carried out for all n-tuples for each component to be purged. The pruned list of n-tuples is sorted so that operators are considered by the ordering algorithm in order of preference. The n-tuples are converted to fully instantiated operator form, which can be used by the ordering algorithm. The ordered list of fully instantiated operators is stored with the appropriate abstract operator. Before pruning, a total of 30 n-tuples is generated for purging O_2, 30 for N_2, and 12 for H_2O (fewer possibilities exist for H_2O because there are fewer choices of methods for purging adsorbed liquids). After pruning, three n-tuples remain for O_2, six for N_2, and four for H_2O. These are listed below, in operator form.

For N_2: Purge N_2 by method sweep with purgative HCl to destination vent-2
Purge N_2 by pressure-up-blow-down with HCl to vent-2
Purge N_2 by sweep with CH_4 to vent-2
Purge N_2 by pressure-up-blow-down with CH_4 to vent-2
Purge N_2 by vacuum to vent-1
(i.e., the material is discharged to the atmosphere)
Purge N_2 by vacuum to vent-2
(i.e., the material is discharged to the vent header)

For O_2: Purge O_2 by method sweep with purgative N_2 to destination vent-1
Purge O_2 by pressure-up-blow-down with N_2 to vent-1
Purge O_2 by vacuum to vent-1

For H_2O: Purge H_2O by sweeping with hot CH_4 to vent-2
Purge H_2O by sweeping with hot N_2 to vent-1

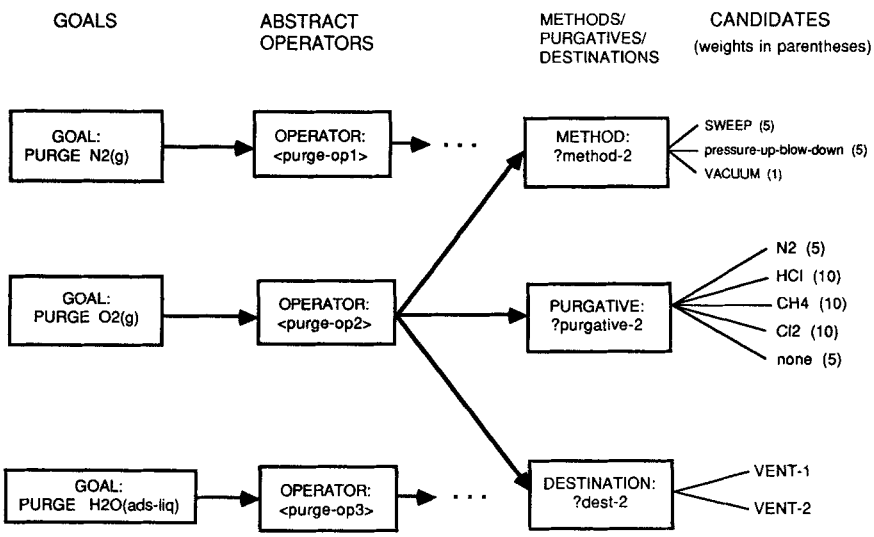

Figure 6. Operator formation for example problem.

Purge H_2O by sweeping with hot N_2 to vent-2
Purge H_2O by sweeping with hot N_2 to drain-1

The next step of planning is to select combinations of operators and to search for orderings of each set. The ordering algorithm, Figure 2, can generate all combinations of one fully instantiated operator from each of the abstract operators (minimal sets) and generate all feasible orderings (if any) of each minimal set. Given the above set of operators, there are 72 minimal sets, and 553 possible unique orderings of one, two, or three operators. The outer backtracking algorithm takes the set of abstract operators, forms minimal sets of fully-instantiated operators, and passes the minimal sets to the inner backtracking algorithm. The outer backtracking algorithm continues to generate minimal sets until either the user indicates that no more alternates are desired, or all possibilities are exhausted.

The inner backtracking algorithm takes a minimal set of operators and seeks to find all feasible orderings of the operators. The inner backtracking algorithm sequences operators according to the depth-first search method described earlier. The operation of the ordering method will be demonstrated with the following set of operators:

pg-op-2: Method: pressure-up-blow-down
Purgative: HCl
Destination: vent-2
Goal: eliminate N_2

pg-op-8: Method: pressure-up-blow-down
Purgative: N_2
Destination: vent-1
Goal: eliminate O_2

pg-op-13: Method: hot-sweep
Purgative: N_2
Destination: drain-1
Goal: eliminate H_2O

The ordering process, shown graphically in Figure 7, begins by forming a planning path containing a mode called "PLANHEAD," representing the initial state. (The initial state vector is shown on the left side of Figure 4.)

The first operator to be proposed to extend the planning path is pg-op-2 (Pressure-up-blow-down with HCl to vent-2). The first step in checking the feasibility of this operator is to check the local constraints for opening the connection between the system and the destination. It is found that local constraint lcl-constr-3 is violated. This constraint forbids opening vent-2 (the hydrocarbons vent header) if O_2 is present in the system. The next operator to be proposed is pg-op-8 (Pressure-up-blow-down with N_2 to vent-1). No local constraints are violated, as N_2 and O_2 are allowed in vent-1 (vent to the atmosphere). The next step in determining the feasibility of this operator is to evaluate the global constraints under conditions estimated to occur during performance of the task. The "conditions-during" list for method pressure-up-blow-down contains the information that the purgative is present in high concentration, and that the system pressure is medium-high (at the extreme). The state vector during performance of the task is:

Temperature = low
Pressure = med-high
Bulk-flow = 0
$[N_{2(g)}]$ = high
$[O_{2(g)}]$ = med
$[H_2O_{(ads\text{-}liq)}]$ = low

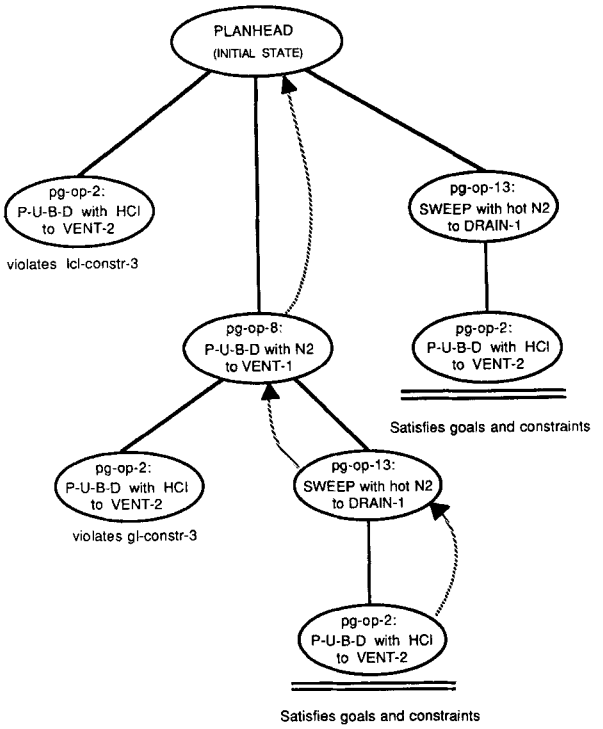

Figure 7. Trace of ordering a set of operators.

Note in this state vector that the effect of dilution of O_2 is not shown. This is because only a rough estimation of the values is needed to evaluate the constraints. No infeasibilities are detected at this hypothetical state. The final step in determining the feasibility of the operator is to evaluate global constraints under conditions estimated after completion of the operation. The "conditions-after" attribute for the method contains the information that the system pressure is low and that only the purgative is present in the gas phase, in high concentration. The resulting state vector is:

Temperature = low
Pressure = low
Bulk-flow = 0
$[N_{2(g)}]$ = high
$[O_{2(g)}]$ = 0
$[H_2O_{(ads\text{-}liq)}]$ = low

This state is found to be feasible. Operator pg-op-8 is placed on the planning path, and the hypothetical state vector for after the purge operation becomes the current state vector. The goals are examined, and it is found that the goal to eliminate O_2 is satisfied.

Next, operator pg-op-2 (Pressure-up-blow-down with HCl to vent-2) is proposed to extend the planning path. This operator is found to be feasible with respect to all local constraints. Conditions during the operation are estimated with the following state vector.

Temperature = low
Pressure = med-high
Bulk-flow = 0
$[N_{2(g)}]$ = high
$[O_{2(g)}]$ = 0
$[HCl_{(g)}]$ = high
$[H_2O_{(ads\text{-}liq)}]$ = low

This state violated global constraint gl-constr-3, as H_2O and HCl are simultaneously present. Operator pg-op-13 (Hot-sweep with N2 to drain-1) is next proposed and found to be feasible. The planning path is completed with operator pg-op-2, at which point all operators have been ordered and all goals satisfied.

At this point, POPS asks if the user desires to find another ordering of this set of operators. When the user replies in the affirmative, the program backtracks by popping pg-op-2 off the path, indicated by the hatched arrow lines in Figure 7. As no untried operators are available at this point on the path, pg-op-13 is removed. No untried operators are available after pg-op-8, so the program backtracks on the PLANHEAD node. Operator pg-op-13 is proposed to extend the planning path, and is found to be feasible. Upon examining the goals, it is found that the goal "eliminate O_2" is satisfied as well as the goal "eliminate H_2O," for which the operator was proposed. Next, the operator pg-op-2 is found to be feasible and added to the path. This satisfies the goal "eliminate O_2." As all three purge goals are satisfied, the planning path is complete, despite the fact that not all of the available operators have been used. No other orderings can be generated for these operators. Note that in this example, POPS did not monotonically pursue the satisfaction of the goal to eliminate N_2. Rather, indirect paths were taken, in which N_2 was used as a purgative before being purged.

The ordering process is performed for all 72 minimal sets. A total of 22 unique, feasible two-operator plans and 60 three-operator plans are generated. All plans are at the "macroscopic" level of purge operators. The next phase of planning, not yet automated in POPS, is to continue the planning at lower levels in the hierarchy of detail, for example, at the level of unit operations and/or the level of valve operations.

Discussion

This paper presents an application of our research in operating procedure synthesis to the problem of purging a chemical processing system during start-up. We identify key decisions to be made (purgative, method, destination) and present a least-commitment planning approach that allows all feasible purge tasks to be identified. We also discuss a qualitative modeling technique, which, together with the method of planning with constraints, allows feasible orderings of tasks to be generated. This work represents an extension of the POPS planning methodology previously reported (Fusillo and Powers, 1987). Some of these extensions are necessary because the domain of purge operations planning involves a very large search space. For the example problem in this paper, there is initially a total of 68,094 possible sequences of purge operations (30 operators each for N_2 and O_2, 12 operators for H_2O in one-, two-, and three-step orderings). After application of consistency constraints during generation of n-tuples, a total of 553 possible sequences remain. Application of local and global system constraints during ordering leaves 82 feasible two- and three-step task sequences. The sets of operators are generated in order of preference, according to the weights expressed in the M, P, D rules. These weights provide a heuristic screening of the purge sequences.

The other extensions to the POPS planning methodology presented here involve the discovery and satisfaction of goals. Constraints are used to discover the goals to eliminate undesirable chemical species. This is analogous to "subgoaling" in other planning methodologies (Newell and Simon, 1972). Finally, the automatic planning system recognizes that a goal has been satisfied, so that redundant tasks are not generated.

Acknowledgment

This research was supported in part by a grant from the Exxon Educational Foundation, and by the National Science Foundation, Grant No. DMC-8616889. The authors also thank William Wrbican for writing assistance.

Appendix

Purge method selection rules

pmselect-rule1
- IF: The component to be purged is gas-phase
- and The required final concentration is extremely low (i.e., on the order of parts per million)
- THEN: Propose purge method *vacuum* with weight = 1

pmselect-rule2
- IF: The component to be purged is gas-phase
- and The system is predominantly liquid-phase
- THEN: Propose purge method = *liquid-displacement* with weight = 1

pmselect-rule3
- IF: The component to be purged is gas-phase
- and The system can act as a pressure bottle
- THEN: Propose purge method = *pressure-up-blow-down* with weight = 5

pmselect-rule4
- IF: The component to be purged is gas-phase
- THEN: Propose purge method = *sweep* with weight = 5

pmselect-rule5
- IF: The component to be purged is adsorbed-liquid-phase
- and Its boiling point is low to medium
- THEN: Propose purge method = *hot-sweep* (i.e., sweep with a hot gas) with weight = 10

pmselect-rule6
- IF: The component to be purged is adsorbed-liquid-phase
- and Its boiling point is high
- THEN: Propose purge method = *absorb* with weight = 5

pmselect-rule7
- IF: The component to be purged is adsorbed-liquid-phase
- and Its boiling point is high
- THEN: Propose purge method = *mechanical* (i.e., "mopping up") with weight = 5

pmselect-rule8
- IF: The component to be purged is liquid-phase
- THEN: Propose purge method = *drain* with weight = 10

pmselect-rule9
- IF: The component to be purged is liquid-phase
- THEN: Propose purge method = *pump* with weight = 5

Purgative selection rules

purgative-select-rule1
  IF: The component to be purged is gas-phase
  THEN: Propose as purgatives all gas-phase components to be present in system at goal state whose concentrations will be ≥ medium
  with weight = 10

purgative-select-rule2
  IF: The component to be purged is gas-phase
  THEN: Propose any available gas-phase purgatives
  with weight = 5

purgative-select-rule3
  IF: The component to be purged is gas-phase
  THEN: Propose as purgatives all liquid-phase components to be present in system at goal state whose concentrations will be ≥ medium
  with weight = 10

purgative-select-rule4
  IF: The component to be purged is gas-phase
  THEN: Propose any available liquid-phase purgatives
  with weight = 5

purgative-select-rule5
  IF: The component to be purged is adsorbed-liquid-phase
  THEN: Propose as purgatives all gas-phase components to be present in system at goal state whose concentrations will be ≥ medium
  with weight = 10

purgative-select-rule6
  IF: The component to be purged is adsorbed-liquid-phase
  THEN: Propose any available gas-phase purgatives
  with weight = 5

purgative-select-rule7
  IF: The component to be purged is gas-phase
  THEN: Propose as purgative *none* (included for compatibility with purge method *vacuum*)
  with weight = 5

Literature Cited

Austin, G. T., *Shreve's Chemical Process Industries*, 5th ed., McGraw-Hill, New York, 759 (1984).

Fusillo, R. H., and G. J. Powers, "A Synthesis Method for Chemical Plant Operating Procedures," *Comput. Chem. Eng.*, **11**, 369 (1987).

Ivanov, V. A., V. V. Kafarov, V. L. Kafarov, and A. A. Reznichenko, "On Algorithmization of the Startup of Chemical Productions," *Eng. Cybernetics*, **18**, 104 (1980a).

———, "Design Principles for Chemical Production Startup Algorithms," *Automation and Remote Control*, **41**, 1023 (1980b).

Kinoshita, A., T. Umeda, and E. O'Shima, "An Algorithm for Synthesis of Operational Sequences of Chemical Processing Plants," *Computerized Control and Operation of Chemical Plants, Vienna, Austria, September, 1981*, Osterreichisher Chemiker, Vienna (1981).

Newell A. and H. A. Simon, *Human Problem Solving*, Prentice Hall, Englewood Cliffs, NJ, 414 (1972).

O'Shima, E., "Safety Supervision of Valve Operations," *J. Chem. Eng. Japan*, **11**, 390 (1978).

Rivas, J. R., and D. F. Rudd, "Synthesis of Failure-Safe Operations," *AIChE J.*, **20**, 320 (1974).

Sacerdoti, E. D., *A Structure for Plans and Behavior*, Elsevier North-Holland, New York (1977).

Stefik, M. J., "Planning with Constraints (MOLGEN, Pt. 1)," *Artific. Intell.*, **16**, 111 (1981).

Stephenopoulos, G., "The Scope of Artificial Intelligence in Plant-wide Operations," FOCAPO Conf. Utah (July, 1987).

Manuscript received May 12, 1987, and revision received on Oct. 19, 1987.