

# Automating Workflow for Data Processing in Grid Architecture

Zhichun Xiao  
Axiom Corporation  
Little Rock, Arkansas 72211  
Email: xiaozc@gmail.com

Craig W. Thompson  
CSCE department  
University of Arkansas  
Fayetteville, Arkansas 72701  
Email: cwt@uark.edu

Wing Ning Li  
CSCE department  
University of Arkansas  
Fayetteville, Arkansas 72701  
Email: wingning@uark.edu

**Abstract**—Workflows are at the heart of data processing solutions. Because of the poor performance and the license cost, traditional relational database management systems are no longer good choices for processing huge amount of data. Nowadays many data processing companies turn to the workflow grid to process huge amount of data. Currently, the generation of workflow is made by human being, which is very tedious, labor intensive, and error prone, and hence becomes the bottleneck of the performance of data processing in the grid architecture. This paper proposes a workflow automation strategy that can replace the human being by a workflow generator that can automatically generate a workflow from given input. A prototype has been implemented and a simulation has been designed.

**Key Words:** Workflow automation, data grid.

## I. INTRODUCTION

Traditionally, relational database management systems (RDBMS) are widely used to build data product. As the data size grows larger and larger, the RDBMS is no longer a good choice because of the poor performance and expensive storage. Nowadays, data processing companies tend to use Grid computing as a replacement of RDBMS because of the high performance and the low cost.

A Grid is a set of commodity computers that are connected together to form a parallel and distributed system. Those computers can be geographically distributed and there is no centralized resource management system to control the coordination of resources [9], [17]. It has been shown that Grid computing can provide nontrivial quality of service at an affordable cost [8], [7], [5].

The Grid architecture can be used to do data processing. In this computation model, data is evenly distributed among the Grid nodes. A set of operators are also distributed on some nodes. A workflow is constructed for each data processing job. When the system is at a heavy load, many workflows run concurrently in the data Grid. Data flow from data grid to a workflow that consists of a sequence of operators. Each operator sends requests to data Grid, gets results back, and sends data to following operators. The performance are guaranteed by two kinds of parallelism: pipeline parallelism and partition parallelism. Many workflows are pipelined to run concurrently within the data Grid. The data are evenly distributed among data Grid nodes so that many queries can be processed simultaneously. See Figure 1 for an illustration.

Data processing in the grid architecture relies heavily on the workflows. A data-processing workflow has three components: primary inputs, primary outputs, and a set of operators (transforms). An operator or transform is an entity that transforms a set of input fields to a set of output fields. Data flows from data grid to a workflow that consists of operators and flows back to data grid again. The operators within the workflow enhance the data by querying the data grid.

With the performance improvement of the workflow grid and data grid, the generation of the workflow itself becomes a bottleneck of the overall performance. Currently, the workflow is generated by human beings. Given a set of primary inputs, a set of primary outputs, and a set of operators, a data expert looks for a set of operators that can transform the primary inputs to the primary outputs and connects them correctly to form a workflow. Figure 2 illustrates the generation of a workflow.

This process requires a data expert to know the detailed information of each operator and data dependencies within the operators. It is not efficient, tedious, and error prone. Moreover, for the same input and output, different data experts may generate different workflows. Thus the result may not be consistent. Therefore, an automated workflow generation solution is demanded and it will greatly improve the overall performance of the data processing in grid architecture. Many papers has been published on the workflow generation and analysis [4], [10], [16], [12], [1], [2], [3], [6].

The benefits of a workflow automation system are obvious. Redundant steps can be found by the workflow automation system and will be eliminated, which will greatly simplify the entire processing system [14]. The number of errors can be dramatically reduced with the help of automatic error checking system [11]. The customer service can be greatly enhanced by providing a higher throughput capabilities and a decreased turnaround time [13]. The training of new personnel on the workflow system is made much simpler [15]. The entire system is more flexible to the changes and improvements.

This paper proposes a workflow automation strategy to automatically generate a workflow in the grid architecture. It is a backtracking based exhaustive search algorithm that explores the solution space and finds the operators, connects the input fields and output fields of the operators, and generates the

workflow. Section 2 gives a formal description of the problem to be solved. The formal description is based on the abstraction of a real business problem that is investigated. Section 3 introduces the algorithm in detail. Section 4 describes the system architecture framework for workflow automation. Section 5 considers the future work and concludes the paper.

## II. PROBLEM DESCRIPTION

The workflow automation problem can be described as follows. To simplify the discussion, a field is represented as an integer in the range of 1 to  $n$ . The set of fields  $F$  is denoted as  $F = \{1, 2, \dots, n\}$ . The primary input  $PI$  is denoted as  $PI = \{a_1, \dots, a_l\}$ . The primary output  $PO$  is denoted as  $PO = \{b_1, \dots, b_k\}$ . Here  $l$  and  $k$  are the number of primary inputs and outputs respectively.

An operator consists of a set of inputs and a set of outputs. A finite set of operators are numbered from 1 to  $m$ . The input of an operator  $T_i$  is denoted by  $I(T_i) = \{I_{i_1}, I_{i_2}, \dots, I_{i_k}\}$  and output of an operator  $T_i$  is denoted by  $O(T_i) = \{O_{i_1}, O_{i_2}, \dots, O_{i_l}\}$ .

Each input, either in the primary input or of an operator, is mapped to a field, which is referred to as the input field. The input fields of an operator are distinct. So are the output fields. Let  $f$  denote the mapping function that maps fields to inputs and outputs. The domain of  $f$  is defined as  $domain(f) = \{\{I_{i_j} \mid 1 \leq i \leq m, 1 \leq j \leq k_i\} \cup \{O_{i_j} \mid 1 \leq i \leq m, 1 \leq j \leq l_i\} \cup \{a_1, a_2, \dots, a_l\} \cup \{b_1, b_2, \dots, b_k\}\}$ . The range of  $f$  is defined as  $range(f) = F$ . The following conditions are satisfied by  $f$ .

- for  $1 \leq i \leq m$ ,  $f(I_{i_j}) \neq f(I_{i_k})$  iff  $j \neq k$
- for  $1 \leq i \leq m$ ,  $f(O_{i_j}) \neq f(O_{i_k})$  iff  $j \neq k$
- for  $1 \leq i, j \leq l$ ,  $f(a_i) \neq f(a_j)$  iff  $i \neq j$
- for  $1 \leq i, j \leq k$ ,  $f(b_i) \neq f(b_j)$  iff  $i \neq j$

Given a set of input fields or output fields  $S$ ,  $f(S) = \{f(x) \mid x \in S\}$ .

Given primary input  $PI$ , primary output  $PO$ , and a set of operators  $OP = \{T_1, T_2, \dots, T_p\}$ , the input set  $I$  of  $PI, PO$

and  $OP$  is defined as  $I = \{b_1, \dots, b_k\} \cup \{I_{i_j} \mid 1 \leq i \leq m, 1 \leq j \leq k_i\}$ . The output set  $O$  is defined as  $O = \{a_1, \dots, a_l\} \cup \{O_{i_j} \mid 1 \leq i \leq m, 1 \leq j \leq l_i\}$ .

A connection (workflow) is a subset of  $I \times O$  (ordered pairs of input and output). A valid connection (valid workflow)  $C$  is a connection that satisfies the first four conditions of the following. A complete connection  $CC$  is a connection that satisfies all of the following conditions.

- Equal field property:  $\forall \langle x, y \rangle \in C, f(x) = f(y)$ .
- Complete input property:  $\forall y \in I(\exists x \in O(\langle x, y \rangle \in C))$ .
- Anti-Symmetric property:  $\langle x_1, y_1 \rangle, \langle x_2, y_1 \rangle \in C \Rightarrow x_1 = x_2$ .
- Acyclic property: when operators are viewed as nodes of a graph and elements of  $C$  are viewed as directed edges, the resulting directed graph is acyclic.
- Complete operator output property:  $\forall x \in O \setminus PI, \exists y \in I, \langle x, y \rangle \in C$ .

Notice that it is possible in a valid connection that some of the elements in  $O$  are not connected. We may view these elements as part of primary output field that has not been filtered yet. The given primary output field is the filtered one. In a complete connection all elements are connected.

For the field based workflow automation problem, we are given  $PI, PO$ , and a set of  $m$  operators  $\{T_1, T_2, \dots, T_m\}$ , and are asked to find a valid connection (or a complete connection) for some subsets of the operators if such a connection exists.

Figure 3 illustrates a workflow generated by a workflow generator. The primary input fields are  $PI = \{1, 2, 3, 4\}$ . The primary output fields are  $PO = \{5, 6, 7, 8\}$ . There are totally four operators,  $A, B, C$ , and  $D$ . The mapping function is a one-to-one mapping, i.e., the input field of an operator can only be mapped to the same output field of another operator. For example, the operator  $B$  has three input fields: 9, 3, and 4. The input fields 3 and 4 has no corresponding output fields in other operators. The input field 9 has a corresponding output field 9

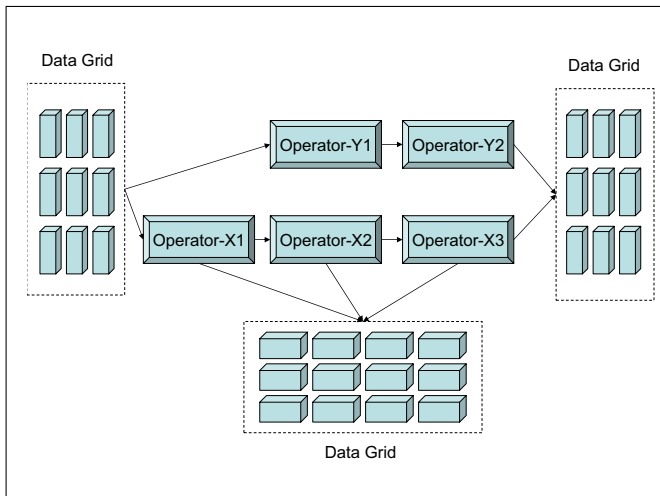


Fig. 1. Data processing workflow in Grid architecture

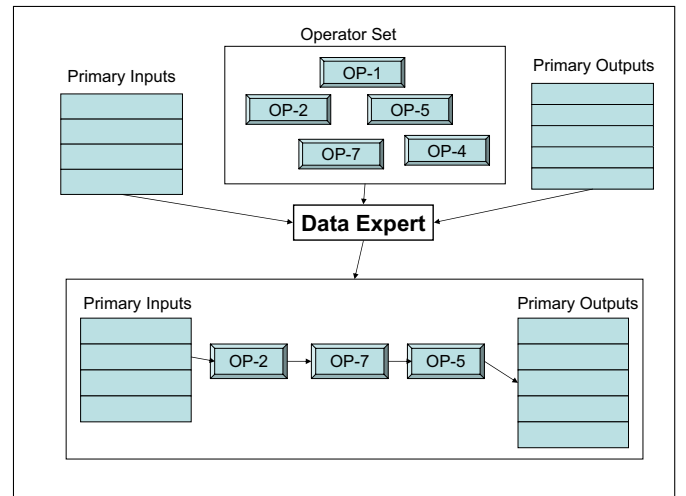


Fig. 2. A workflow generated by data experts

in operator  $A$ . Therefore, operator  $A$  is a necessary predecessor operator for operator  $B$  to be used in a valid workflow. As can be seen in Figure 3, more than one valid workflow can be generated from the given instance.

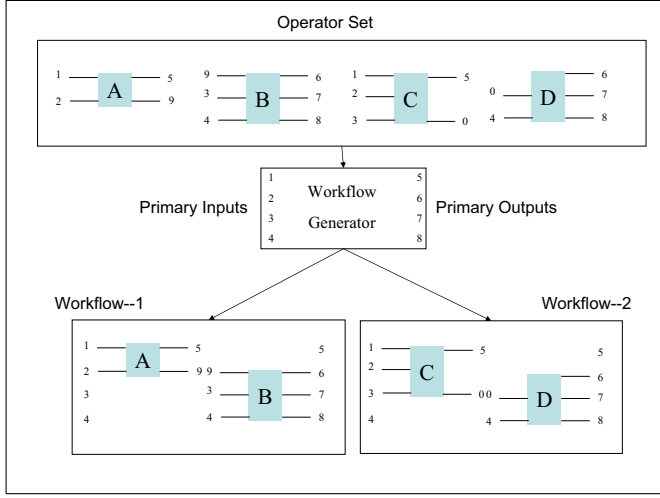


Fig. 3. A Workflow Generation Example

### III. ALGORITHMS

A backtracking based exhaustive search algorithm is proposed to explore the solution space, and hence to solve the valid connection problem. The algorithm consists of two procedures: engine and connect.

- WorkflowEngine() has four parameters: three inputs and one output. The inputs are primary input set, primary output set, and operator set. The output is the  $OP$  set with connection  $C$ .
- connect() also has four parameters: three inputs and one output. The inputs are available output set, desirable output sets, and operator set. The output is  $OP$  set with connection  $C$ . connect is recursive and returns true if a valid connection is found and false otherwise.

**Algorithm 1: Given primary input  $PI$ , primary output  $PO$ , and  $m$  operators, return a workflow.**

```

WorkflowEngine( $PI, PO, m, OP$ )
1  $Y = f(PO) \cap f(PI)$ ;
2  $AO = PI$ ;
3  $DO = \{x \mid x \in PO, f(x) \notin Y\}$ ;
4 if (connect( $AO, DO, m, OP$ ))
5   AddSimpleConnection( $OP, PI, PO, Y$ );
6 else
7   no solution;

```

Fig. 4. Workflow engine.

The set  $Y$  is the intersection of primary output fields and primary input fields. It contains all the fields that flow directly from the initial state to the final state without going through

any operator, thus they need not to be considered in the workflow generation. The set  $AO$  contains the available output fields, which are used for mapping to the input fields of an operator. The set  $DO$  contains the desired output fields, which are all the fields yet to be generated by operators to be added to a workflow.

The WorkflowEngine() calculates the available output  $AO$  and desired output  $DO$ , and call the connect() function to find the necessary operators and connect them together. If all the fields in the desired output are found in the operators, it claims success and then call AddSimpleConnection() function to directly connect the fields in  $Y$  from the primary input  $PI$  to the primary output  $PO$ . On the other hand, if there exists some fields in  $DO$  that cannot be found in the output fields of all operators, it reports error and claim failure.

**Algorithm 2: Given available output  $AO$ , desired output  $DO$ , and  $m$  operators, make a connection.**

```

connect( $AO, DO, m, OP$ )
1 if ( $DO = \emptyset$ ) return true;
2 Let  $T_{i_1}, T_{i_2}, \dots, T_{i_k}$  be those unmarked operators
3 satisfying the following conditions. If none,  $k$  is 0.
4 a)  $f(I(T_{i_j})) \subseteq f(AO), 1 \leq j \leq k$ 
5 b)  $|f(O(T_{i_1})) \cap f(DO)| \geq \dots \geq |f(O(T_{i_k})) \cap f(DO)|$ 
6 c)  $|f(O(T_{i_j})) \cap f(DO)| = |f(O(T_{i_{j+1}})) \cap f(DO)| \Rightarrow$ 
7  $|f(I(T_{i_j})) \cup f(AO)| \geq |f(I(T_{i_{j+1}})) \cup f(AO)|, 1 \leq j \leq k-1$ 
8 for ( $j = 1; j \leq k; j++$ ) {
9   mark  $T_{i_j}$ ;
10   $R = \{x \mid x \in DO, f(x) \in f(O(T_{i_j}))\}$ ;
11  if (connect( $AO \cup O(T_{i_j}), DO \setminus R, m, OP$ )) {
12    AddOperatorConnection( $OP, AO, R, T_{i_j}$ );
13    return true;
14  }
15  else
16    unmark  $T_{i_j}$ ;
17 }
18 return false;

```

Fig. 5. Connection algorithm.

The connect() is a recursive function. It first test if the  $DO$  is empty. If  $DO$  is empty, which means all the fields have been found, it claim success and return. Otherwise, it sort the unmarked operators  $T_{i_1}, T_{i_2}, \dots, T_{i_k}$  in a descending order with respect to the cardinality of the intersection of the output field and the desired output. This greedy scheme can expedite the searching algorithm. The set  $R$  is the intersection of the output field of current marked operator with the desired output. It then call connect() recursively to find the remaining fields in unmarked operators. After all the fields are found successfully, the AddOperatorConnection() is called to connect the necessary operators together to form a workflow.

AddSimpleConnection( $OP, PI, PO, Y$ ) uses field based connection to connect elements from  $PI$  to  $PO$  using fields given by  $Y$  and updates  $OP$  accordingly. Similarly, AddOperatorConnection( $OP, AO, R, T_{i_j}$ ) connects elements from  $AO$  to  $I(T_{i_j})$  and from  $O(T_{i_j})$  to  $R$ . In general, choices may exist to connect  $AO$  to  $T_{i_j}$ . The connection from  $O(T_{i_j})$

to  $R$  is unique.

The time complexity of the above algorithm might be further reduced by introducing one or more bounding functions.

#### IV. SYSTEM ARCHITECTURE DESIGN OF THE WORKFLOW AUTOMATION

We have designed a three level architecture for the workflow automation system. At the highest level, data experts specify the initial fields (primary input) and the final fields (primary output). The operator database and mapping database are given. The operator database specifies the functionality of the operators and the input and output fields of each operator. The mapping database specifies the precedence relationships among operators and the mapping policy of each operator. For some operators without explicit mapping policy, expert orderings are used to finalize the topological order of the operators. The second level is the workflow generator engine. It takes the input from the first level and generates a workflow automatically. The third level is the workflow generated from the workflow engine. Figure 6 shows the current architecture of the workflow automation system.

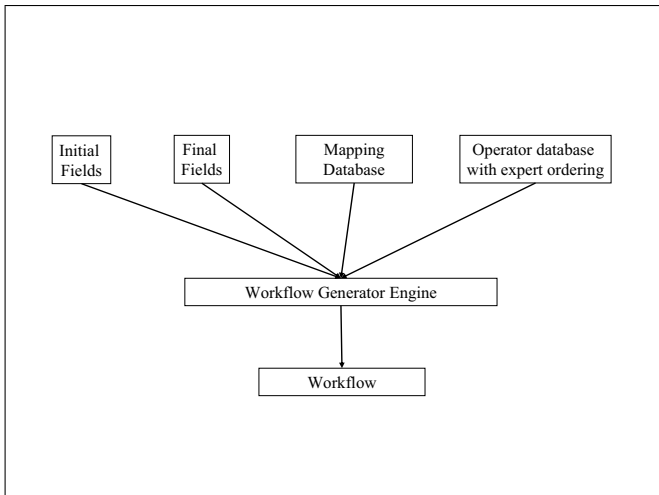


Fig. 6. Current Workflow Automation System Architecture

We also designed a four level architecture to further improve the automation of the workflow generation. The third level is the workflow generation engine, which takes as input the initial fields, final fields, the mapping database, the operator database, and parameter settings, and generates a valid workflow as the output. Although this level itself can generate workflow automatically, it requires the user to master and specify many detailed information for the input parameters of an operator. It is not an easy job when the number of operators become large. Therefore, another business language level is designed to further liberate the users from the tedious work they have to do in the past. With the business language, the user can simply specify the objective of his job in a SQL like language, and the business language interpreter will generate all the data in the second level, which will be feed into the workflow generator engine at the third level. Figure 7 shows the architecture of the system.

#### V. CONCLUSION AND FUTURE WORKS

Grid computing is becoming a trend of the future. More and more companies are using Grid computing to improve the performance of information processing and hence the performance of business. Workflow plays an important role in data processing within the Grid architecture. The generation of a correct and efficient workflow is not an easy job even for an experienced employee. Therefore, automatic workflow generation will greatly improve the overall performance of grid computing. It has been shown that a multi-level workflow automation architecture is applicable both in theory and in reality. The problem of building a workflow with respect to a given way of processing the data file is formulated and a backtracking based search algorithm is proposed to automatically find necessary operators within the operator set and connect them into a workflow.

The current model simplified the workflow automation by ignoring some details in the real world. For example, the cost of the operator is not considered in the current solution. The parameter setting of some operators also need the involvement of human being. These factors will be investigated in the future.

#### ACKNOWLEDGMENT

The authors would like to thank Computer Science and Computer Engineering department at University of Arkansas and Axiom Corporation for providing data, infrastructures, and funding to do the research.

#### REFERENCES

- [1] K.R. Abbott and S. K. Sarin. Experiences with workflow management: issues for the next generation. In *CSCW '94 Proceedings*, pages 113–120, Chapel Hill, NC, 1994.
- [2] P. Barthelmeß and J. Wainer. Workflow systems: a few definitions and a few suggestions. In *Proc. of the COOCS' 95*, pages 138–147, Milpitas, CA, 1995.
- [3] D. P. Bogia and S. M. Kaplan. Flexibility and control for dynamic workflows in the worlds environment. In *Proc. of the COOCS' 95*, Milpitas, CA, 1995.

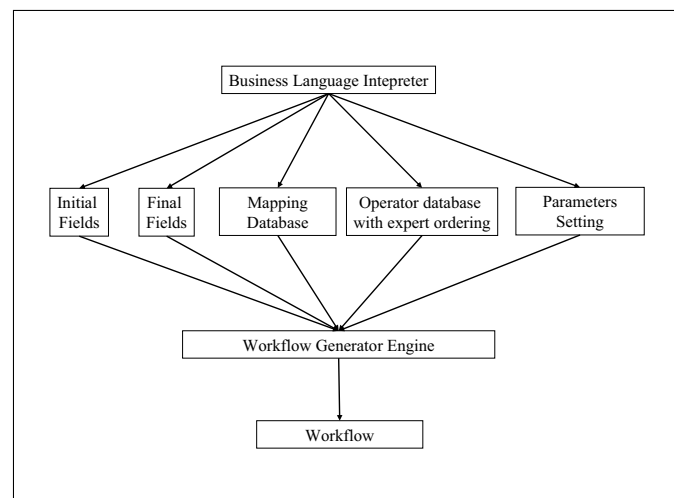


Fig. 7. Workflow Automation System Architecture

- [4] Pat Bowe and Brian O'Flaherty. Implementing workflow automation systems: Implications for management control. In *Proceedings of the 29th Annual Hawaii International Conference on System Sciences - 1996*, pages 13–22, Hawaii, USA, 1996.
- [5] Rajkumar Buyya. *Economic-based Distributed Resource Management and Scheduling for Grid Computing*. PhD thesis, School of Computer Science and Software Engineering, Monash University, Melbourne, Australia, 2002.
- [6] N. Craven and D. Mahling. Goals and process: A task basis for projects and workflows. In *Proc. of the COOCS' 95*, pages 237–248, Milpitas, CA, 1995.
- [7] Ian Foster. The anatomy of the grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150:1, January 2001.
- [8] Ian Foster and Carl Kesselman. *The grid: blueprint for a new computing infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.
- [9] Ian Foster and Carl Kesselman. Computational grids. *Lecture Notes in Computer Science*, 1981:3, January 2001.
- [10] Dimitrios Georgakopoulos, Mark F. Hornick, and Amit P. Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases*, 3(2):119–153, 1995.
- [11] Chris Hallett. Improving customer service with workflow automation at abbey national. In *Proc. OIS Document Management 92 (London, 1992)*, pages 405–409, London, 1992.
- [12] D. W. Judge, B. R. Odgers, J. W. Shepherdson, and Z. Cui. Agent-enhanced workflow. *BT Technology Journal*, 16:79–85, July 1998.
- [13] Steven O. Kimbrough and Scott A. Moore. Message management systems: Concepts, motivations, and strategic effects. *Journal of Management Information Systems*, pages 1–31, May 1992.
- [14] Mandy Lavery. A survey of workflow management software. In *Proc. OIS Document Management 92 (London, 1992)*, pages 398–404, London, 1992.
- [15] L. Schultz. Computer programs and cataloguing: One innovative approach. *OCLC Micro*, 4, 1988.
- [16] Edward A. Stohr and J. Leon Zhao. Workflow automation: Overview and research issues. *Information Systems Frontiers*, 3:281–296, September 2001.
- [17] Yong Zhao, Michael Wilde, Ian T. Foster, Jens-S. Vockler, Thomas Jordan, Elizabeth Quigg, and James Dobson. Grid middleware services for virtual data discovery, composition, and integration. In *Proceedings of the 2nd workshop on Middleware for grid computing - 1996*, pages 57–62, Toronto, Canada, 2004.