Analyzing Data Dependence Based on Workflow Net*

Yu Huang, Hanpin Wang**, Wen Zhao, and Jiaqi Zhu

School of EECS, Peking University, Beijing, China, 100871
huangyu1979@gmail.com, whpxhy@pku.edu.cn

Abstract. Workflow management systems (WfMSs) frequently use data to coordinate the execution of workflow instances. A WfMS makes routing decisions according to data constraints. This paper presents an extended workflow net which has its business form and portrays data constraints among tasks which act as the decision condition. Data constraint is expressed by pre-condition and post-condition which formalize a task's input and output conditions. After introducing the model, we give an important definition of consistence related to data dependance. Then, we proposes a domain computation method during reduction to verify the consistence. The method considers data operation during reduction. In the end, a case study is analyzed by our method.

1 Introduction

Workflow has become an essential technology in e-business, providing a flexible and appropriate environment for development and maintenance of next-generation component-oriented information systems for enterprise applications [1]. Nowadays, workflow management systems have been widely deployed in the domains of administration, production, and scientific research in order to ensure high efficiency of business process.

There exist many formal models mainly aiming at control flow and resource perspective, e.g. [1][2][3], but few concerning data perspective. Data perspective mainly handles business documents and other objects which flow between activities, and local variables of the workflow. Data perspective is crucial for the correctness of a workflow execution, because routing decisions depend on it. There are a series of concepts that apply to the representation and utilization of data within workflow systems. [4] integrated workflow model with production data model to support production development. [5] proposed data guard to guarantee significant changes in data. Nevertheless, no articles have given formal models linking the business form with data dependence and verification methods to check the soundness related to data.

* Supported by the National Grand Fundamental Research 973 Program of China under Grant No. 2002CB312004 and 2002CB312006 and the National 863 Plans Projects of China under Grant No. 2006AA01Z160.

** Corresponding author.

Y. Shi et al. (Eds.): ICCS 2007, Part III, LNCS 4489, pp. 257–264, 2007.

[2] proposed a subclass of Petri nets called the workflow net (WF-net) which models the control flow perspective of a workflow. Tasks are represented by transitions, and procedure logic is modeled by places and arcs. Some papers extended WF-net to model resource perspective, critical section, etc. [3][7]. This paper gives an extended workflow net describing the data dependence. Then we introduce the inconsistent problem after adding data constraint. Therefore, we give the definition of data consistency. Furthermore, we propose a new method to verify its property.

The structure of the remainder is: section 2 states basic definitions about workflow and its extension. Section 3 discusses the conditional expressions and some new reduction rules for analyzing data consistency. This paper concludes in section 4.

2 Basic Workflow Definitions

In this section, we give the basic notions of workflow net and its extension.

Definition 1 (Petri net). *A Petri net is a triple* (P, T, F) *where:*

- P *is a finite set of places.*
- T *is a finite set of transitions, and* $P \cap T = \varnothing$.
- $F \subseteq (P \times T) \cup (T \times P)$ *is a set of edges.*

Given a Petri net (P, T, F), we will use the following notations: ${}^\bullet p = \{t| < t, p > \in F\}, p^\bullet = \{t| < p, t > \in F\}, {}^\bullet t = \{p| < p, t > \in F\}, t^\bullet = \{p| < t, p > \in F\}$.

Definition 2 (Workflow net (WF-net)). *A Petri net* $N = (P, T, F)$ *is a WF-net if and only if:*

- N *has two special places:* i *and* o *where* ${}^\bullet i = \varnothing$, $o^\bullet = \varnothing$.
- *If we add a transition* t^* *to* N, *connecting place* o *with* i *so that* ${}^\bullet t^* = i$, *the Petri net* N^* *obtained is strongly connected. where* t^* *represents a transition which connects the output to the input.*

Definition 3 (Soundness). *A WF-net* $N = (P, T, F)$ *is sound if and only if:*

- *For every state or marking* M *reachable from state* i, *there exists a firing sequence leading from state* M *to state* o. *Formally:*
  $\forall_M (i \xrightarrow{*} M) \Rightarrow (M \xrightarrow{*} o)$
- *State* o *is the only state reachable from state* i. *Formally:*
  $\forall_M (i \xrightarrow{*} M \wedge M \geq o) \Rightarrow (M = o)$
- *There are no dead transitions in (N,i). Formally:*
  $\forall_{t \in T} \exists_{M, M'} i \xrightarrow{*} M \xrightarrow{t} M'$

Examples of the business process include insurance claim, visa application, goods ordering and goods transportation, etc. All these processes share three qualities: case-driven, process-essential and a predefined explicit procedure. Every process has a predefined (or predesigned) business form(B-form) or document, like a

visa application form, an order form of goods. Therefore, B-form is the key to define a workflow process. A B-form(document) is a view for an information set required to support a task which is the unit consisting of a workflow. A B-form is filled up and its content is changed when it is passed from task to task.

Definition 4 (B-form). *B-form is a set of typed variables which have been evaluated.*

Definition 5 (Domain Expression). *Domain expression is the following form:* $\{(variable_1, variable_2, \cdots, variable_n), (valueRange_1, valueRange_2, \cdots, valueRange_n)\}, n \in \mathbb{N}$. *E represents that all variables can be evaluated into arbitrary values which are allowed in their type.* $\varnothing$ *denotes that all variables cannot be valuated into any values.*

Domain expression is divided into two parts: one represents the variables, the other represents the respective evaluated ranges.

Definition 6 (Data Constraint). *The data constraint of a task indicates its precondition and postcondition. When a transition is fired, the B-form in this moment must satisfy its data constraint. Data constraint is composed of two parts: data constraint=*{*precondition, postcondition*}.

The precondition and postcondition are represented by domain expressions which describe the constraint of variables in the B-form. The variables which aren't included in the precondition or postcondition represent that the fired transition does not care them. We call $\{precondition, postcondition\}$ bi-domain.

Definition 7 (Data Dependence Workflow net (DDWF-net)). *A DDWF-net is Five tuples* $N = (P, T, F; \varphi, B)$, *satisfying the followings:*

1. *(P,T,F) is a WF-net.*
2. *B is a B-form which is dealt with in the workflow. The initial values in B are null. And* B_0 *represents the initial B-form which means initial marking* $M_0 = \{i = B_0\}$.
3. φ *is a function from tasks to the set of data constraints.*

Figure 1 is an example of **application of leaving** in government affair processing. In the DDWF-net, B-form is represented by token. The variables in B-form is changed or copied by tasks which can be fired. Since every task has data constraint, there exists consistency between the B-form and data constraint. When

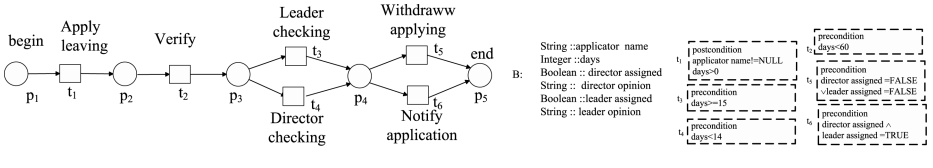

Fig. 1. An Example of DDWF-net

a task can be fired, we must evaluate whether the B-form satisfies its precondition. For example, in Figure 1, if t_2 can be fired and $days > 60$, then there exists conflict with the constraint of t_2. What is more, at the point of choice or parallel, we must judge whether the preconditions of following tasks have the relationship of choice or parallel. Actually, the decision-making must be consistent with the control relation. If p_3 has one token and the $days = 14$, dead state occurs. Therefore, we must give one method to solve these problems. Next, we define the property of consistency.

Definition 8 (Data Consistency of DDWF-net). *A DDWF-net* $N = (P, T, F; \varphi, B)$ *is consistent, satisfying the followings:*

1. *For every marking* M *which can be reachable from* i*, if every* p *in* P *and* $M(p) \neq 0$*, domain of every variable in B-form of* $M(p)$ *is in the union of respective variables in preconditions of* $p^\bullet$*.*
2. *For every place* p*, the intersection of every variable of preconditions in* $p^\bullet$ *is empty.*

According to the definition, place p_2 in Figure 1 dissatisfies item 2, since the intersection of variable days in the preconditions of t_3, t_4 is not empty. Therefore, data consistency is an import property.

3 Domain Computation Analysis Based on Reduction

Workflow process model is a description and specification model of business process. Structural and data conflict of process model is hard to be detected without computer aid. Therefore, an automatic mechanism is essential to discover errors in the model quickly, and the analysis of workflow has gradually become a hot yet difficult spot in the current workflow research. In [2], Aalst WMP made use of the analytical method of Petri nets to verify the soundness of Wokflow net(WF-net). Many scholars share in common with the idea of reduction which can be used to detect conflicts in models. The earliest is Sadiq who proposed several reduction rules to reduce workflow graph(acyclic graph) for detecting structural conflict[1]. However, there are no researches based on data dependence. How to verify the data consistency of a workflow model is a complex problem. In the following, we introduce a domain computation method based on reduction.

Definition 9 (Domain Operation). *Domain operation gives all the operators in domain expression.*

1. *divide(/): a/b means removing the variables in b from a and the result acts as the return of a/b.*
2. *symmetry substraction(*$\otimes$*):*$a \otimes b$ *return the common variables and the commutative substraction of their ranges. For example:* $((x, y), (\{1, 2\}, \{2, 3\})) \otimes ((x, z), (\{0, 2\}, \{1, 3\})) = ((x), (\{0, 1\}))$.

3. *intersect($\cap$): $a \cap b$ return not only the common variables and the intersection of their ranges but also the different variables. For example:* $((x, y), (\{1, 2\}, \{2, 3\})) \cap ((x, z), (\{0, 2\}, \{1, 3\})) = ((x, y, z), (\{2\}, \{2, 3\}, \{1, 3\}))$.
4. *union ($\cup$): $a \cup b$ return not only the common variables and the union of their ranges but also the different variables. For example:* $((x, y), (\{1, 2\}, \{2, 3\})) \cup ((x, z), (\{0, 2\}, \{1, 3\})) = ((x, y, z), (\{0, 1, 2\}, \{2, 3\}, \{1, 3\}))$.

The intersect and union satisfy commutative laws and distributive laws:

1. $a \cup b = b \cup a$, $a \cap b = b \cap a$.
2. $(a \cup b) \cap c = (a \cup c) \cap (b \cup c)$.
3. $(a \cap b) \cup c = (a \cap c) \cup (b \cap c)$.

In order to relate data constraint to control flow, we define some conditional expressions to describe the computation of data constraint. We use the form $(pre_1, post_1)$ to represent data constraint.

Definition 10 (conditional expression(bi-domain computation)). *First, we introduce the basic operators.*

1. *OR:* $(pre_1, post_1) \vee (pre_1, post_2) = ((pre_1 \cup pre_2), post_1)$ *if* $post_1 = post_2$. *Its semantics is if two elements have choice relationship and the same postcondition then the unitary precondition is the union of their preconditions.*
2. *AND:* $(pre_1, post_1) \wedge (pre_2, post_2) = ((pre_1, post_1 \cup post_2))$ *if* $pre_1 = pre_2$. *Its semantics is if two elements have parallel relationship and the same precondition then the unitary postcondition is the union of their postconditions.*
3. *ADD:* $(pre_1, post_1) + (pre_2, post_2) = (pre_1 \cap (pre_2/post_1), post2 \cap (post_1/(pre_2 \cup post_2))$ *if* $post_1 \subseteq pre_2$ *that means the range of each variable in* $post_1$ *is a subset of the rang of respective variable in* pre_2. *This operation expresses the simple connection. If* $\neg post_1 \subseteq pre_2$, *then the expression would return to an error because of the occurrence of dead lock. Therefore, through estimating the condition, we can judge whether the model has the property of data consistency.*

Then, we introduce distributive laws in operators.

1. $((pre_1, post_1) \vee (pre_2, post_2) \cdots \vee (pre_n, post_n)) + (pre, post) = ((pre_1 \cap (pre/post_1)) \cup (pre_2 \cap (pre/post_2)) \ldots \cup (pre_n \cap (pre/post_n)), post \cap ((post_1/(pre \cup post)) \cup (post_2/(pre \cup post)) \cdots\cdots (post_n/(pre \cup post))))$ *if* $post_1 \cup post_2 \cup \cdots post_n \subseteq pre$, *otherwise* $\neg post_1 \cup post_2 \cup \cdots post_n \subseteq pre$ *would lead to deadlock and an error would be thrown. Therefore, according to this operator, we can judge the data consistency.*
2. $((pre_1, post_1) \wedge (pre_2, post_2) \cdots \wedge (pre_n, post_n)) + (pre, post) = (pre_1 \cap pre_2 \cdots \cap pre_n \cap pre/post_1 \cap pre/post_2 \cdots \cap pre/post_n, post \cap (post_1/(pre \cup post)) \cap (post_2/(pre \cup post)) \cdots \cap (post_n/(pre \cup post)))$ *if* $post_1, post_2, \cdots post_n \subseteq pre$, *otherwise dead lock would occur and an error would be thrown which dissatisfies the data consistency.*

3. $(pre, post) + ((pre_1, post_1) \vee (pre_2, post_2) \cdots \vee (pre_n, post_n)) = ((pre \cap (pre_1/post)) \cup (pre \cap (pre_2/post)) \cdots \cup (pre \cap (pre_n/post)), post_1 \cap ((post/(pre_1 \cup post_1)) \cup (post_2 \cap (post/(pre_2 \cup post_2))) \cdots \cup (post_n \cap (post/(pre_n \cup post_n)))))$ *if* $pre_1 \cup pre_2 \cup \cdots pre_n \subseteq post$ *and* $\forall i, j, pre_i \otimes pre_j = E(i \neq j)$*, otherwise dead lock would occur and an error would be thrown which dissatisfies the data consistency.*
4. $(pre, post) + ((pre_1, post_1) \wedge (pre_2, post_2) \cdots \wedge (pre_n, post_n)) = (pre \cap pre_1/post \cap pre_2/post \cdots \cap pre_n/post, post_1 \cap post_2 \cdots \cap post_n \cap (post/(pre_1 \cup post_1)) \cap (post/(pre_2 \cup post_2)) \cdots \cap (post/(pre_n \cup post_n)))$*if* $post_1, post_2, \cdots post_n \subseteq$ *pre, otherwise dead lock would occur and an error would be thrown which dissatisfies the data consistency.*

Theorem 1 (Expression Consistency). *If a conditional expression can be computed to the form of one bi-domain and the precondition of the respective bi-domain is E, then the expression is data consistent. And during computing, if an error is thrown, the expression is data inconsistent.*

Because all computations of expressions keep the data consistency, the theorem naturally comes into existence.

After introducing the conditional expression or domain computation, we must solve the problem of how to construct the expression in order to compute whether the DDWF-net has consistent data constraints. We use the reduction rules to build the conditional expression. The following reduction rules keep soundness in control flow based on [1]. For the simpleness of reducing procedure, every place also has the (pre,post) constraint. In the beginning, the constraint of every place is (E, E) representing no restriction, i.e., all variables can be their typed values.

Reduction 1. *The sequential structure which is composed of two places and one transition can be reduced into a* ADD *conditional expression. Figure 2 shows the procedure.*

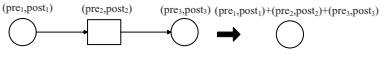

Fig. 2. Rule One

Fig. 3. Rule Two

Reduction 2. *The sequential structure which is composed of two transitions and one place can be reduced to a ADD conditional expression. Figure 3 shows the procedure.*

Reduction 3. *This rule describes the merging of xor-split and xor-join. Figure 4 shows the procedure.*

Reduction 4. *This rule describes the merging of and-split and and-join. Figure 5 shows the procedure.*

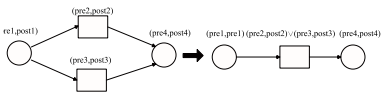

Fig. 4. Rule Three

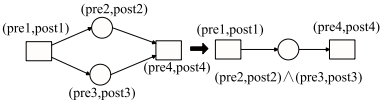

Fig. 5. Rule Four

Theorem 2 (Soundness and Data Consistency). *If a DDWF-net can be reduced to one place without any error and the precondition of the respective bi-domain is E, then DDWF-net is data consistent.*

According to [1], that one model can be reduced to one place means the model is sound. Due to theorem 1, that the conditional expression can be computed to one bi-domain denotes the model is data consistent. Therefore, this theorem comes into existence.

We give a DDWF-net which is sound but data inconsistent through applying reduction rules. Figure 6 shows the example. Its B-form has only three variables $x \in \{1,2,3,4,5,6\}$,$y \in \{0,1,2,3\}$,$z \in [0,1]$. And the respective data constraints are: $g_1 = (E,E), g_2 = (x > 2, E), g_3 = (x \leq 2, E), g_4 = (y > 2 \wedge x > 4, E), g_5 = (y \neq 3, E), t_1 = (E, (z > 0.5)), t_2 = (E, (y > 2)), t_3 = (E, (x > 4)), t_4 = (E, (z < 0.3 \wedge y = 3)), t_5 = (E, (x > 4))t_6 = (E, (z > 0.2)), t_7 = (E, E)$, other places are equal to (E, E).

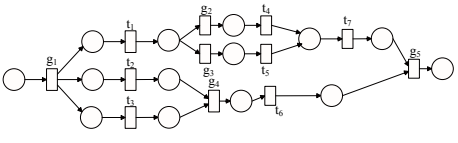

Fig. 6. DDWF-net with Data Inconsistent

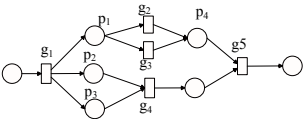

Fig. 7. DDWF-net After First Reduction

Furthermore, we construct the conditional expression through the reduction rules. Firstly, we apply the rule one and two to construct conditional expression which is shown in Figure 7. The respective data constraints are: $p_1 = t_1, p_2 = t_2, p_3 = t_3, p_4 = t_7, g_2 = (x > 2, (z < 0.3 \wedge y = 3)), g_3 = (x \leq 2, (x > 4)), g_4 := (y > 2 \wedge x > 4, (z > 0.2))$,others remain intact. Secondly, we apply the rule three and four shown Figure 8. The respective data constraints are: $g_6 = g_2 \vee g_3 = (x > 2, (z < 0.3 \wedge y = 3)) \vee (x \leq 2, (x > 4)), p_5 = p_2 \wedge p_3$. Thirdly, Figure 9 shows the third reduction. The respective data constraints are: $p_6 = p_1 + g_6 + p_4 = (E, (z > 0.5)) + ((x > 2, (z < 0.3 \wedge y = 3)) \vee (x \leq 2, (x > 4))) + (E, E) = (E, (z < 0.3 \wedge y = 3 \wedge x > 4)), p_7 = p_5 + g_4 = (E, (z > 0.2))$. At last, we reduce the model into one place, get one conditional expression, and judge whether it can be computed into one bi-domain. The conditional expression is $g_1 + (p_6 \wedge p_7) + g_5$. It is obvious that $(p_6 \wedge p_7) + g_5$ can throw an error. Actually, postcondition of $(p_6 \wedge p_7)$ satisfies $y = 3$, but the precondition of g_5 dissatisfies $y = 3$. Though the example is small, it shows that our method is correct and usable.

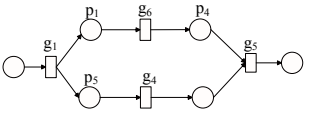

Fig. 8. DDWF-net After Second Reduction

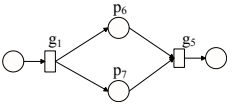

Fig. 9. DDWF-net After Third Reduction

4 Conclusion

This paper presents an extended workflow net which can describes business form and data constraints among tasks acting as the decision conditions. Data constraint is expressed by bi-domain, and we give an important definition of consistency related to data dependence. Then, we propose some domain operators and conditional expressions. During the reduction, we construct the conditional expression in order to analyze the data consistency. This paper also analyzes an example to confirm our methods. For further study, we will do more extended work on verification of our workflow model and implement a tool based on our method.

References

1. Wasim Sadiq, Maria E. Orlowska. Applying Graph Reduction Techniques for Identifying Structural Conflicts in Process Models. In M.Jarke and A. Oberweis, editors, Proceedings of the 11th International Conference on Advanced Information Systems Engineering (CAiSE '99), *Lecture Notes in Computer Science* vol. 1626: 195-209. Springer-Verlag, Berlin, 1999.
2. W.M.P. van der Aalst, The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems and Computers*, vol. 8, pp. 21-66, 1998.
3. Kees van Hee, Alexander Serebrenik, Natalia Sidorova, and Marc Voorhoeve, Soundness of Resource-Constrained Workflow Nets. *Lecture Notes in Computer Science* vol. 3536, pp. 250-267, 2005.
4. Nigel Baker, Alain Bazan, G. Chevenier, Florida Estrella, Zsolt Kovacs, T. Le Flour, Jean-Marie Le Goff, S. Lieunard, Richard McClatchey, Steve Murray, Jean-Pierre Vialle, An Object Model for Product and Workflow Data Management. *Proceedings of Ninth International Workshop on Database and Expert Systems Applications*, pp. 731-738, 1998.
5. Johann Eder and Marek Lehmann, Workflow Data Guards, *Lecture Notes in Computer Science* vol. 3760, pp. 502-519, 2005.
6. Bartosz Kiepuszewski. Expressiveness and Suitability of Languages for Control Flow Modelling in Workflows. PHD thesis, 2002.
7. Yehia Thabet Kotb, Andre Stephan Baumgart. An Extendted Petri net for modeling workflow with critical secitons. *Proceedings of the 2005 IEEE International Conference on e-Business Engineering.*
8. Jaeho Kim, Woojong Suh, Heeseok Lee. Document-based workflow modeling: a case-based reasoning approach. *Expert Systems with Applications* vol. 23, pp. 77-93, 2002.