Performance Optimization of Temporal Reasoning for Grid Workflows Using Relaxed Region Analysis

Ke Xu[1], Junwei Cao[2,3,*], Lianchen Liu[1,3] and Cheng Wu[1,3]
[1]National CIMS Engineering and Research Center, Tsinghua University, Beijing, 100084, P. R. China
[2]Research Institute of Information Technology, Tsinghua University, Beijing, 100084, P. R. China
[3]Tsinghua National Laboratory for Information Science and Technology, Beijing, 100084, P. R. China
*Corresponding email: jcao@tsinghua.edu.cn

Abstract

With quick evolution of grid technologies and increasing complexity of e-Science applications, reasoning temporal properties of grid workflows to ensure reliability and trustworthiness is becoming a critical issue. Relaxed Region Analysis (RRA) is proposed in this work for performance optimization of grid workflow verification by decomposing workflows into separate standard regions with parallel branches. The approach is implemented in GridPiAnalyzer, a Pi Calculus based formal verifier for grid workflows, and validated using gravitational wave data analysis workflows. Detailed experimental results illustrate that RRA can dramatically reduce CPU and memory usage of verification processes.

1. Introduction

The grid is becoming a mainstream technology for cross-domain management and sharing of computational resources [6]. Grid workflows [2, 15], a composition of various grid services according to prospective processes, have become a typical paradigm for problem solving in various e-Science domains, e.g. gravitational wave data analysis [5].

With increasing complexity of e-Science applications, how to implement reliable and trustworthy grid workflows according to specific scientific criteria is becoming a critical research issue. In addition to existing grid *enabling* techniques, e.g. job scheduling, workflow enactment and resource locating, various grid *ensuring* techniques are developed [12], e.g. data flow analysis and temporal reasoning. While these techniques aim to guarantee that large scale grid workflows can be developed to meet exact requirements of domain-specific users, performance is still a bottleneck for probing all potential pitfalls and errors in large scale and dynamically evolving grid workflows. Implementation of grid verification processes has to be of high performance in terms of CPU and memory usage.

Performance optimization of formal verification of grid workflows is focused in this work. Following our preliminary efforts on grid workflow decomposition [13], a Relaxed Region Analysis (RRA) approach is proposed to divide-and-conquer global verification of a grid workflow into local verification on sub grid workflow models. Target grid workflows are decomposed into sequentially composed regions with relaxation of parallel workflow branches. The approach is implemented in *GridPiAnalyzer* [14], a Pi Calculus [8] based formal verifier for grid workflows using NuSMV2 [3] as its engine. Three application scenarios of using workflow technologies for gravitational wave data analysis are investigated [1]. While the complexity of a grid workflow increases exponentially with the number of involving services and interdependencies, the RRA approach can dramatically reduce CPU and memory usage of formal verification processes, as illustrated using quantitative performance evaluation results included in this work.

Formal verification based temporal reasoning [4] is becoming essential for Web Services based systems to probe potential errors and enhance reliability. How process algebras can be applied to model and reason the choreography of Web Services is discussed in [10]. Regarding grid system formalization, the Abstract State Machine based formalism is applied in [9] to distinguish grid features from traditional distributed systems. In our previous work described in [12, 14], a formal framework is proposed as an integrated solution to reliability issues in existing grid applications.

Decomposition is a common technique for handling complex systems to exponentially decrease system dimensions. While application-specific decomposition strategies have been investigated in [11] for carrying out computational tasks in grid environments, a more general decomposition approach is proposed in our work for grid workflow verification using process structure analysis. The RRA approach described in this paper is a follow-up work of our initial decomposition efforts in [13]. It allows the relaxation of parallel branches in grid workflows to achieve better decomposition results and verification performance.

The rest of the paper is organized as follows. In Section 2, grid workflow regions are defined. Section 3 introduces the RRA approach and how it works in the

decomposed verification strategy. The implementation of RRA in *GridPiAnalyzer* and corresponding performance evaluation results are given in Section 4. Section 5 concludes the paper.

2. Grid Workflow Regions

Considering that there are various grid workflow specification languages, common notations used in this paper are provided in Figure 1 to visually represent a grid workflow model. Modeling elements in Figure 1 are extended from typical Directed Acyclic Graph (DAG) based workflow models, allowing explicit modeling of data service nodes, service control nodes, conditional transitions and arbitrary cycles of transitions. While DAG is already well-defined and intuitive, these extensions are more expressive to cover many other existing workflow specifications.

To prevent unstructured grid workflows, syntactical constraints are defined as a unified basis for our region analysis. These constraints are concluded from sound criteria, e.g. no deadlocks and no multiple service activity instances on the same service activity.

Constraint 1: We refer a *Srv&Ctrl* node to a *Grid Service Activity*, *Subflow* or *Control* node and refer a *SrvFlow* node to a *Srv&Ctrl* or *Data Service* node.

Constraint 2: Each grid workflow has exactly one explicit *Begin* node and *End* node (which will be later relaxed in our RRA approach).

Constraint 3: Every *Srv&Ctrl* node must be syntactically reachable from the *Begin* node and can reach the *End* node by transitions (i.e., no dangling *Grid Service Activity*, *Subflow*, or *Control* nodes).

Constraint 4: Each transition has exactly one source / target *Srv&Ctrl* node. Each data channel has at most one source / target *SrvFlow* node (with one of them must be a *Data Service* Node).

Constraint 5: Multiple inputs and outputs are allowed for a *Grid Service Activity* and *Control* node. Their equivalent semantics are illustrated in Figure 1(b).

Constraint 6: Arbitrary cycles are allowed as long as no unstructured workflow models are caused.

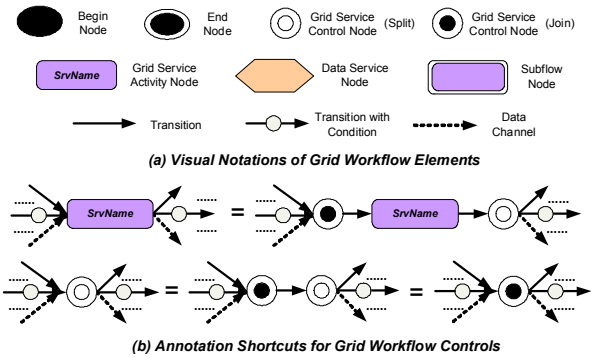


Figure 1. Visualization of grid workflow elements

Figure 2 illustrates an example gravitational wave data analysis workflow *SF1* based on visual notations provided in Figure 1.

A *region* $\{N_{head}, N_{tail}\}$ specifies a structure in which node N_{head} will always reach N_{tail} in order for it to reach the *End* node in a grid workflow Γ (and vice versa). For example, in Figure 2 {*TrigBank_H2_3, thIncaII_L1H2*} is a region while {*sInca_L1H1, thIncaII_L1H2*} is not. The whole grid workflow Γ itself also forms a region. A node *N'* is thus said to be *within* a region $\{N_1, N_2\}$ (denote by $N' \subset \{N_1, N_2\}$) if there exists a path $N_1 \rightarrow \ldots \rightarrow N' \rightarrow \ldots \rightarrow N_2$. Two nodes N_{head} and N_{tail} form a *maximized region* in a grid workflow Γ, if and only if (IFF) $\forall Begin \rightarrow N_1 \rightarrow \ldots N_m \rightarrow End$ where N_{head} and N_{tail} are contained in the path and $N_{head} \neq N_{tail}$.

Moreover, for nodes $N'_1, \ldots N'_m$ within region $\{N_1, N_2\}$, the set of maximized regions $\{\{N', N''\} \mid \{N', N''\} = \{N_1, N'_1\}$ or $\{N'_1, N'_2\}$ or … or $\{N'_m, N_2\}\}$ is said to be a *total decomposition* of $\{N_1, N_2\}$ IFF all $\{N', N''\}$s are maximized regions and can not be decomposed further. A maximized region $\{N_1, N_2\}$ in Γ is a *standard region* IFF $\{N_1, N_2\}$ belongs to the total decomposition of Γ. A standard region will always exist for Γ (in the worst case the only standard region will be Γ itself). For example in Figure 2, while {*TrigBank_H2_3, thIncaII_L1H2*} is a region, it is neither a maximized region nor a standard region. However, {*Begin, Inspiral_L1*} is a standard region of Γ. Figure 2 also shows standard regions of *SF1*.

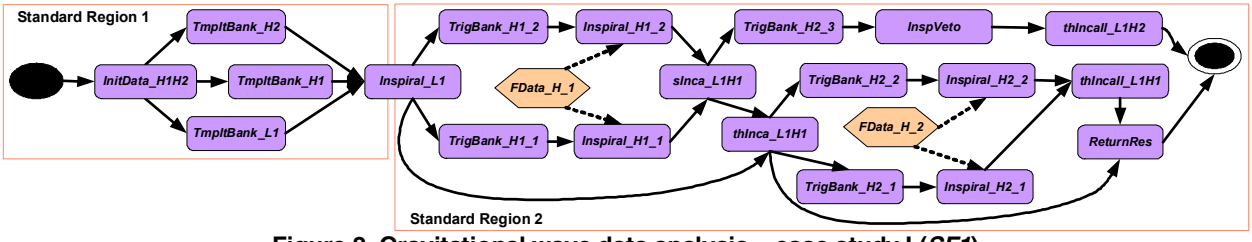


Figure 2. Gravitational wave data analysis – case study I (*SF1*)

3. Grid Workflow Decomposition

3.1. Standard Region Analysis

Apart from the decomposition of grid workflows, the decomposition of corresponding formal verification strategy has also to be developed, which includes:

(1) How to exploit the properties of a standard region into its verification;
(2) How to exploit local verification of a standard region into verification of other standard regions;
(3) How to deduct the global verification result based on local verification of standard regions.

Above issues can be actually transformed into a special modular model checking problem [7]. As we know, the idea of formal verification is to find all states $\{s \in M \mid M,s \models f\}$, where M is the state model [4] (e.g. kripke structure, automata, etc) of the target system and f is the desired property. It is said that M satisfies f (i.e. $M \models f$) if the set of states s is not empty. A modular model checking tries to deduct the formal verification procedure in the following form:

$$\frac{<TRUE>M<\varphi> \quad <\varphi>M'<\psi>}{<TRUE>M \mid M'<\psi>} \qquad \text{(d-1)}$$

The deduction tries to prove that if model M satisfies property φ ($<TRUE>M<\varphi>$) and model M' satisfies property ψ under the *assumption* that its environment satisfies property φ ($<\varphi>M'<\psi>$), the parallel composition of ($M|M'$) will satisfy property ψ ($<TRUE>M|M'<\psi>$). An essential procedure in the above deduction is how to define and implement $<\varphi>M'<\psi>$ such that the deduction will hold true. Consequently, our decomposition strategy of verifications based on standard regions follows the idea below: given the total decomposition $\{M_1, M_2, ..., M_n\}$ of a grid workflow Γ where $M_i=\{N_i, N_{i+1}\}$, $N_i, N_{i+1} \in \Gamma$, the verification of a desired property ψ is carried out on $M_n,...,M_1$ separately, whereas the verification of M_i against ψ will be based on the satisfaction of $M_{i+1};...;M_n$ against ψ such that the satisfaction of the complete workflow Γ against ψ can be eventually deducted.

$$\frac{<TRUE>M_{i+1};.....;M_n<\psi_{i+1}> \quad <\psi_{i+1}>M_i<\psi_i>}{<TRUE>M_i;M_{i+1};.....;M_n<\psi_i>} \qquad \text{(d-2)}$$

Here we have $1 \le i \le n-1$ and $M;M'$ indicates the sequential composition of identified standard regions since sequential relations are preserved among standard regions. The following takes LTL-X (a popular temporal logic with universal path qualifiers and no *next* operators) [4] as the target for the implementation of $<\varphi>M'<\psi>$ (i.e. both φ and ψ are specified in LTL-X). LTL-X is an intuitive and shuttering closed logic with wide formal verification tool support. Since an important theoretical foundation is that LTL-X formulae can be transformed to an equivalent generalized büchi automata [4], $<\varphi>M'<\psi>$ can be obtained by verifying $Trans(\varphi)|M' \models \psi$ [7], where $Trans(\varphi)$ indicates the equivalent automata for φ. However in this work, the sequential nature of standard regions enables us to further avoid the cost of automata composition.

Given the total decomposition $\{M_1, M_2, ..., M_n\}$ of a grid workflow Γ where $M_i=\{N_i, N_{i+1}\}$, $N_i, N_{i+1} \in \Gamma$, denote $TransSys(\Gamma, \Phi)$ to be the automata for Γ under the given initial state set of Φ. Since M_i and M_{i+1} share the service node N_{i+1}, the set of *association states* $Im(M_i, M_{i+1}, \Gamma)$ is the states when $M_i;M_{i+1};...;M_n$ transits to the process of $M_{i+1};M_{i+2};...;M_n$.

The association states literally indicate the *region initial states* for the previous local verification ($<TRUE>M_{i+1};M_{i+2};...;M_n<\psi_{i+1}>$) ($S(M_{i+1};M_{i+2};...;M_n)$) and the *region ending states* for the current local verification ($<\psi_{i+1}>M_i<\psi_i>$) ($\mathbb{E}(M_i)$) in the deduction procedure (d-2).

In the total decomposition of Γ, the only shared states of the corresponding automata for standard regions M_i and M_{i+1} are their association states. This implies that no states in one standard region will loop back to states in another standard region.

An important decomposition strategy for formal verification based on standard regions is, given a standard region M_i in a grid workflow Γ, the desired LTL-X formula Ψ and its sub formulae $\varphi \in sub(\Psi)$, if $<TRUE>M_{i+1};M_{i+2};...;M_n<\varphi>$ holds, we can deduct the satisfaction of $<TRUE>M_i;M_{i+1};M_{i+2};...;M_n<\varphi>$ by investigating whether $(Trans(M_i, S(M_i;...;M_n));Trans(\varphi)), S(M_i;...;M_n) \models \varphi$ holds. Here ";" represents the sequential composition of $Trans(M_i, S(M_i;...;M_n))$ and $Trans(\varphi)$.

3.2. Relaxed Region Analysis

One deficiency of the above workflow decomposition using standard regions is that it has imposed strong constraints (see Section 2) on grid workflow structure analysis, which sometimes limits verification performance since identified standard regions may not small enough. For example in the decomposition result in Figure 2, the identified standard region {*Inspiral_L1*, *End*} can be still considered as a complex sub workflow.

One of key factors in decomposing the verification of a grid workflow Γ is to assure that $TransSys(M_i,S(M_i;...;M_n)) \supseteq S(M_{i+1};M_{i+2};...;M_n)$ s.t. the sequential composition of $M_i;...;M_n$ will not loose

complete behaviors in the original grid workflow. Under this condition, it is inspired to relax Constraint 2 in Section 2 to allow multiple *End* nodes in Γ such that potential parallel branches can also be discovered in addition to sequential standard regions.

Relaxation of Constraint 2: Each grid workflow has exactly one explicit *Begin* node and can be relaxed to allow multiple *End* nodes. New *End* nodes after relaxation are named secondary end nodes (*VEnd*).

For a standard region, denote $M_i/(N_1 \rightarrow ... \rightarrow N_m)$ $(N_j \in M_i,\ j=1,...,m)$ as the operation of removing a branch in Γ with corresponding grid workflow nodes, transitions and data channels. It is then expected to find more potential standard regions in a grid workflow by temporarily removing a selected branch, and to make the verification decomposition result still work in this relaxed context.

A parallel branch $CP=N_1 \rightarrow *N_2 \rightarrow \ldots \rightarrow VEnd$ indicates a path that ends with *VEnd*, such that a parallel composition relation holds between grid service nodes in $N_2 \rightarrow \ldots \rightarrow VEnd$ and new discovered standard regions after node N_1 (i.e. there are no control/data constraints in service executions among them). In the total decomposition $\{M'_1, M'_2, ..., M'_m\}$ of $M_n/(N_2 \rightarrow \ldots \rightarrow VEnd)$, if $N_1 \in M'_i$, it is called that the parallel branch CP belongs to M'_i, denoted by $M'_i(CP)$. $\{M_1, M_2, ..., M'_1, M'_2, ..., M'_i(CP), ..., M'_m\}$ is therefore called the relaxed total decomposition after the relaxation of $CP=N_1 \rightarrow N_2 \rightarrow \ldots \rightarrow VEnd$ for the grid workflow Γ.

Denote $CP'=N_2 \rightarrow \ldots \rightarrow VEnd$, because CP' forms a parallel relation with all the rest standard regions $(M'_{i+1}, ..., M'_m)$ when $M'_i(CP)$, we have $Trans((M'_k, ..., M'_m | CP')) \supseteq Trans((M'_{k+1}, ..., M'_m | CP'))$ for any $i \le k < m$ under the same initial state *Init*. Therefore the verification under relaxed region analysis can also reuse the results in Section 3.1. The whole deduction procedure includes 4 steps, as shown in (d-3):

$$\begin{cases} \dfrac{<TRUE>(M'_{j+1};.....;M'_m)|\mathbb{CP}'<\psi'_{j+1}> \quad <\psi'_{j+1}>M'_j|\mathbb{CP}'<\psi'_j>}{<TRUE>(M'_j;M'_{j+1};.....;M'_m)|\mathbb{CP}<\psi_i>} & i<j\le m \\ \dfrac{<TRUE>(M'_{i+1};.....;M'_m)|\mathbb{CP}'<\psi'_{i+1}> \quad <\psi_{i+1}>M'_i(\mathbb{CP}')<\psi'_i>}{<TRUE>(M'_i;M'_{i+1};.....;M'_m)|\mathbb{CP}'<\psi_i>} & \\ \dfrac{<TRUE>M'_{l+1};.....;M'_m<\psi'_{l+1}> \quad <\psi'_{l+1}>M'_l<\psi'_l>}{<TRUE>M'_l;M'_{l+1};.....;M'_m<\psi'_l>} & 1\le l\le i-1 \\ \dfrac{<TRUE>M_{k+1};.....;M_{n-1};M'_1;.....;M'_m<\psi_{k+1}> \quad <\psi_{k+1}>M_k<\psi_k>}{<TRUE>M_k;M_{k+1};.....;M_{n-1};M'_1;.....;M'_m<\psi_k>} & 1\le k\le n \end{cases}$$

(d-3)

The former two in (d-3) represent cases of standard regions with consideration of parallel branches, while the latter two are used to deal with normal situations described in Section 3.1. The RRA flow chart is illustrated in Figure 3, which is based on the *TotalDecomposition* algorithm.

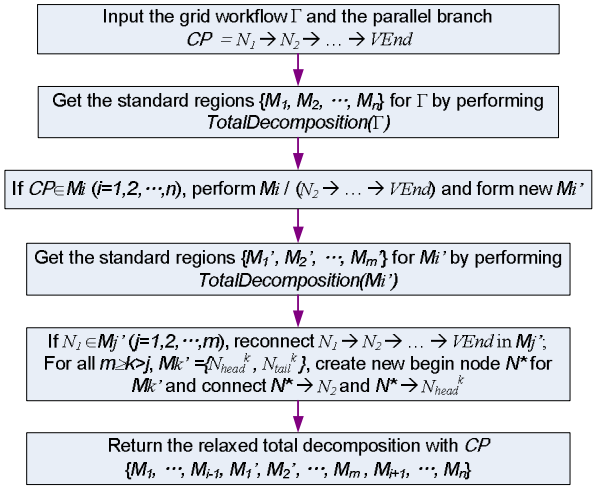


Figure 3. The RRA flow chart

In the gravitational wave workflow *SF1*, since *sInca_L1H1→*TrigBank_H2_3→InspVeto→thIncaII_L1H2→VEnd* is a parallel branch, the original standard region {*Inspiral_L1*, *End*} can be further decomposed into 2 smaller regions, {*Inspiral_L1*, *thInca_L1H1*} and {*thInca_L1H1*, *End*} based on the RRA approach (see Figure 4).

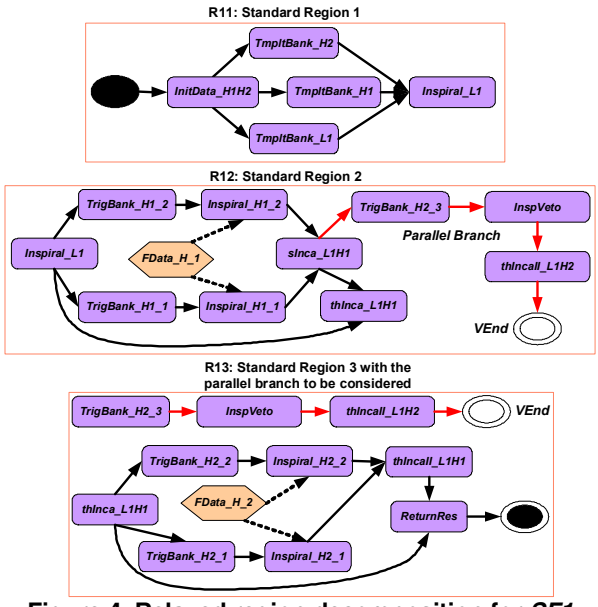


Figure 4. Relaxed region decomposition for *SF1*

4. System Implementation

4.1. GridPiAnalyzer

The RRA approach with corresponding verification strategy described in Section 3 is implemented in our *GridPiAnalyzer* system to further improve its performance in reasoning grid workflows.

GridPiAnalyzer is an automatic analyzer designed for ensuring reliability of grid workflow based on its Pi calculus formalism and verification.

GridPiAnalyzer accepts target grid workflow scripts, e.g. DAG specifications, BPEL4WS and UML activity diagrams. It automatically transforms grid workflow specifications into the process algebra of Pi Calculus and deduces the result into labeled transition systems according to the operational semantics of Pi calculus. A visual environment is provided to specify required temporal properties on grid workflows using LTL-X. These formulae, together with the transition system, are accepted to perform the formal verification.

GridPiAnalyzer is further extended with the capability of our RRA approach for performance optimization. A new component is developed to decompose grid workflows into standard regions with parallel branches based on the procedure described in Figure 3. The formal verification is then recursively performed on each standard region instead of the whole grid workflow. Detailed information about *GridPiAnalyzer* and its applications can also be found in our previous work [12-14].

4.2. Case Studies

Let's take gravitational wave data analysis as our application scenarios. Along with the relatively simple workflow *SF1* introduced in Figure 2, two more complex ones *SF2* and *SF3* are also given in Figures 5 and 7 for performance evaluation of grid workflow verification, with corresponding relaxed standard regions illustrated in Figures 6 and 8, respectively.

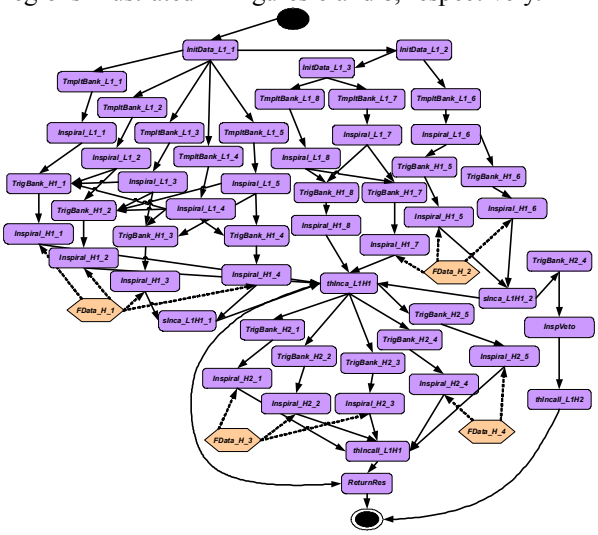


Figure 5. Gravitational wave data analysis – case study II (*SF2*)

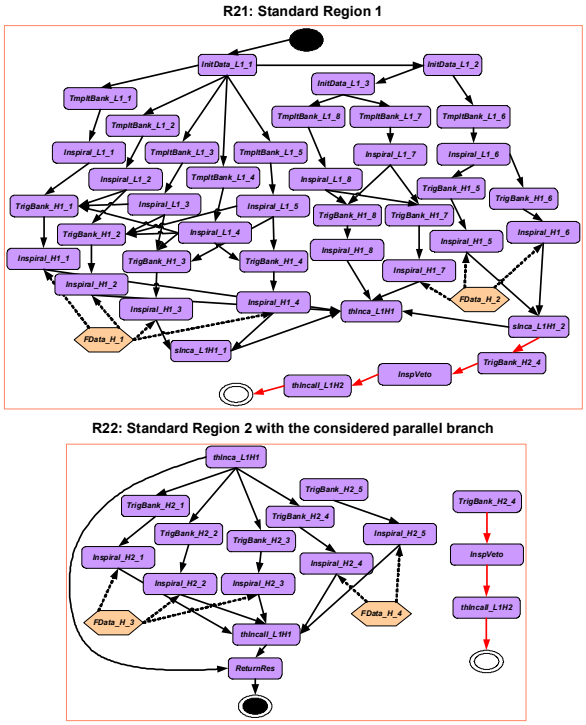


Figure 6. Relaxed region decomposition for *SF2*

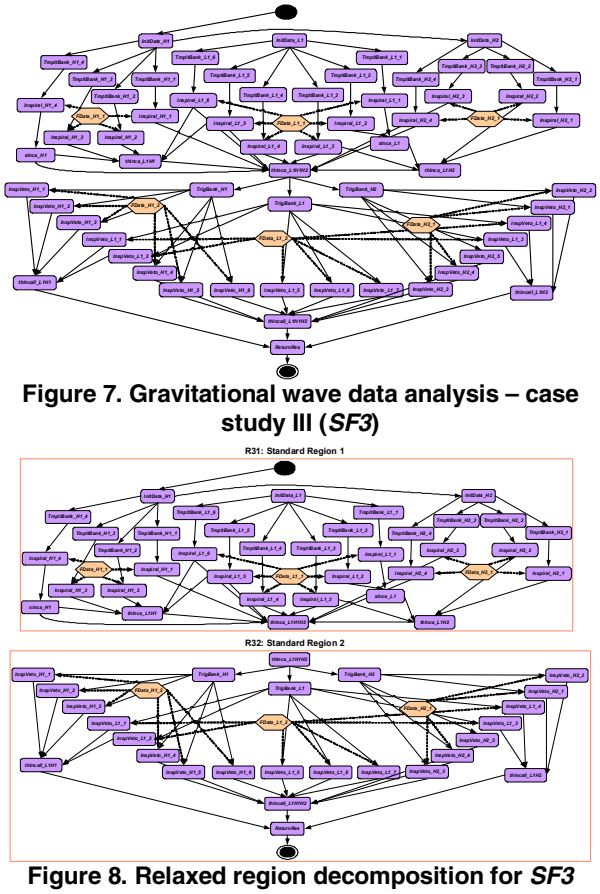


Figure 7. Gravitational wave data analysis – case study III (*SF3*)

Figure 8. Relaxed region decomposition for *SF3*

In gravitational wave data analysis workflows, the following properties are required:

- *p1*: The necessary successor operations after template bank generation;
- *p2*: The constraints on working status of gravitational wave detectors;
- *p3*: The completeness of incidental analysis;
- *p4*: The precondition of final incidental analysis.

Correspondingly the LTL-X formulae of these properties for each grid workflow are formulated in Table 1. These properties are required to be true for all three case studies.

Table 1. Formulae of properties against *SF1*, *SF2* and *SF3*

Prop Name	Target SF	Formulae for the Desired Property
$p1_1$	SF1 / SF3	G (*TmpltBank* _ *H*1 → ((F *TrigBank* _ *H*1) & (F *Inspiral* _ *H*1))
$p1_2$	SF1 / SF3	G (*TmpltBank* _ *H*2 → ((F *TrigBank* _ *H*2) & (F *Inspiral* _ *H*2))
p1	SF2	G (*TmpltBank* _ *L*1 → ((F *TrigBank* _ *H*1) & (F *Inspiral* _ *H*1))
p2	SF1 / SF2	G ((*InitData* _ *H*1\H2) → (¬ *Inspiral* _ *H*2 U *thInca* _ *L*1\H1))
	SF3	G ((*InitData* _ *H*1\H2) → (¬ *Inspiral* _ *H*2 U *thInca* _ *L*1\H1\H2))
p3	SF1 / SF2	((F *sInca* _ *L*1\H1 ∧ (¬*thIncaII* _ *L*1\H1 U *sInca* _ *L*1\H1)) ∨ (F *thInca* _ *L*1\H1 ∧ (¬*thIncaII* _ *L*1\H1 U *thInca* _ *L*1\H1))) ∧ F *thIncaII* _ *L*1\H1
	SF3	((F *sInca* _ *L*1 ∧ (¬*thIncaII* _ *L*1\H1\H2 U *sInca* _ *L*1) ∧ F *sInca* _ *H*1 ∧ (¬*thIncaII* _ *L*1\H1\H2 U *sInca* _ *H*1)) ∨ (F *thInca* _ *L*1\H1\H2 ∧ (¬*thIncaII* _ *L*1\H1\H2 U *thInca* _ *L*1\H1\H2))) ∧ F *thIncaII* _ *L*1\H1\H2
$p4_1$	SF1 / SF2	G (*Inspiral* _ *H*1 → (F *thIncaII* _ *L*1\H1))
	SF3	G (*Inspiral* _ *H*1 → (F *thIncaII* _ *L*1\H1\H2))
$p4_2$	SF1 / SF2	G (*Inspiral* _ *H*2 → (F *thIncaII* _ *L*1\H1))
	SF3	G (*Inspiral* _ *H*2 → (F *thIncaII* _ *L*1\H1\H2))
$p4_3$	SF1	G (*TmpltBank* _ *H*1 → (F *thIncaII* _ *L*1\H1))
	SF2	G (*TmpltBank* _ *L*1 → (F *thIncaII* _ *L*1\H1))
	SF3	G (*TmpltBank* _ *H*1 → (F *thIncaII* _ *L*1\H1\H2.))
$p4_4$	SF1	G (*TmpltBank* _ *H*2 → (F *thIncaII* _ *L*1\H1))
	SF3	G (*TmpltBank* _ *H*2 → (F *thIncaII* _ *L*1\H1\H2))

4.3. Performance Evaluation

In this section, the proposed RRA approach is applied in formal verification of grid workflows and its performance is compared with several well known verification methods [3]. These include the Symbolic Model Checking Algorithm (*SMCA*), *SMCA* with Cone of Influence (*COI*), *SMCA* with dynamic re-ordering of BDD (Binary Decision Diagram) variables (*Dynamic*), and Bounded Model Checking algorithm (*BMC*(*k*), where *k* is the setting of its length). All experiments are carried out using an IBM laptop with a Pentium IV 1.73GHz mobile CPU, 2.0GB RAM, Windows operating system and Eclipse development platform.

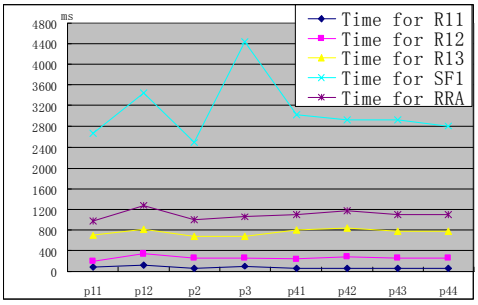
Performance evaluation results are illustrated in Figures 9 and 10 in terms of verification time and peak memory usage, respectively. The upper limit of verification time is 750s for a complete grid workflow and 600s for standard regions. The minimum considered peak memory usage is 10Mb during grid workflow verification. Among legends in Figures 9 and 10, *SMCA'* indicates the time / memory usage for the model initialization with the *SMCA* method. *SF1*, *SF2*, and *SF3* represent the three case studies illustrated in Figures 2, 5 and 7, respectively. R_{ij} indicates the verification on the i^{th} identified standard region in SF_j. *RRA* indicates the verification result using the proposed approach. For a specific property *p*, the total verification time of $RRA(t_p)$ is computed as follows:

$$t_p = \sum_{i=1}^{n} t_p(R_i),$$

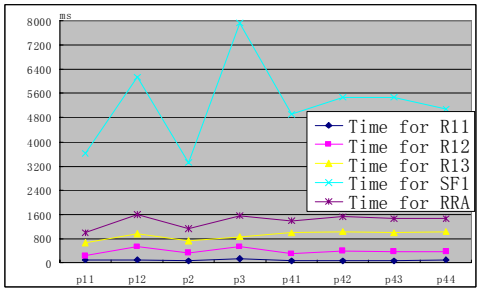
where $t_p(R_i)$ is the verification time for *p* on the i^{th} region in a grid workflow; the peak memory usage of $RRA(m_p)$ is computed as:

$$m_p = Max(m_p(R_i)),$$

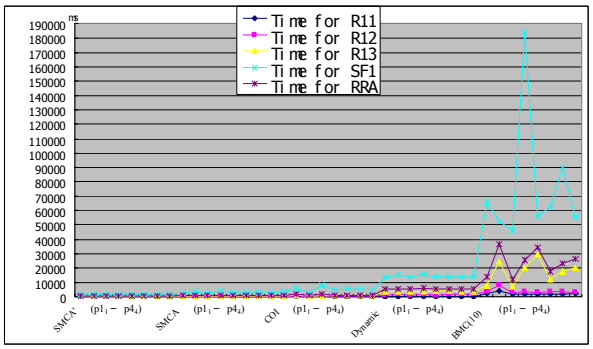
where $m_p(R_i)$ is the peak memory usage for *p* on the i^{th} region in a grid workflow. The purpose of the additional Figures 9(a) and 9(b) is to further zoom in the performance comparison with RRA and pure *SMCA* / *COI* approach in Figure 9(c).



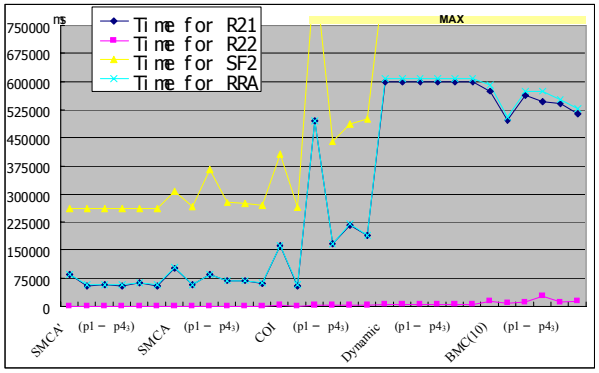
(a)



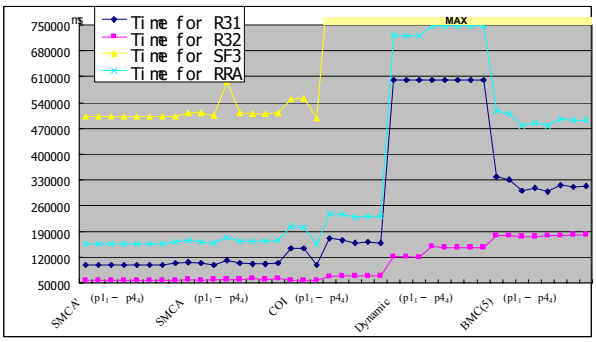
(b)



(c)

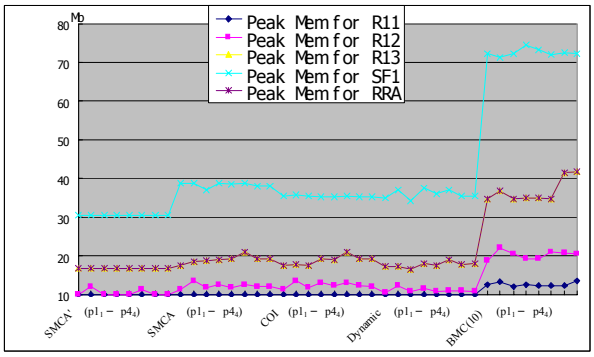


(d)

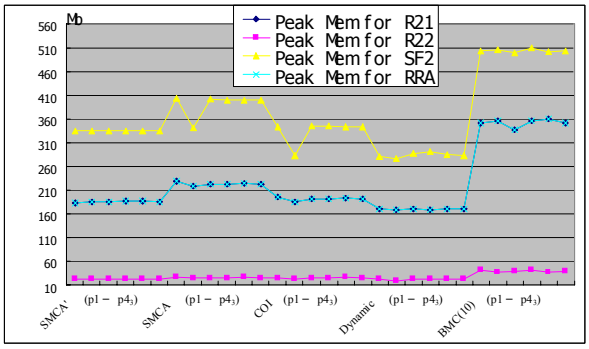


(e)

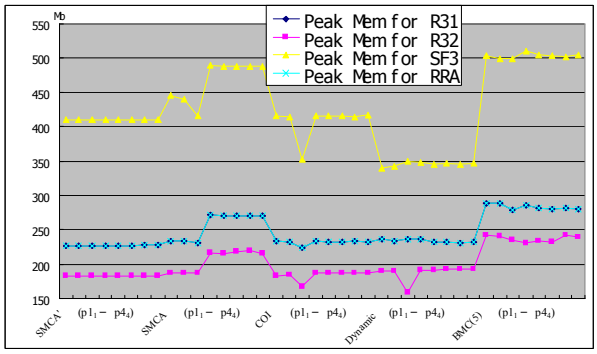
Figure 9. Performance evaluation of verification time for *SF1*, *SF2* and *SF3*



(a)



(b)



(c)

Figure 10. Performance evaluation of perk memory usage for *SF1*, *SF2* and *SF3*

From results included in Figures 9 and 10, it can be found that due to the complexity of grid workflows *SF2* and *SF3* (which contains 2^14.5 and 2^15.4 reachable states, respectively), direct verification of these workflows with none of the compared methods shows satisfactory performance. Based on *SMCA* and *COI*, verification time exceeds 250s and 500s respectively; peak memory usage exceeds 400Mb and 480Mb respectively. Based on *Dynamic* and *BMC*, verification time even exceeds the upper limit and peak memory assumption exceeds 290Mb, 510Mb, 349Mb and 511Mb, respectively. Poor performance of the *BMC* approach is partially due to the use of the simple Mini-SAT solver. Performance comparison between *BMC* and *SMCA* is out of the scope of this paper.

Based on our proposed RRA approach, the verification time is reduced to over 60s, 160s, 590s and 480s for *SF2* and *SF3* in combination with the *SMCA*, *COI* and *BMC* method respectively. The peak memory usage is dramatically reduced to 230Mb, 196Mb, 171Mb, and 360Mb for *SF2* and 270Mb, 232Mb, 236Mb, 290Mb for *SF3*. Here for RRA with the *Dynamic* method, the verification time for the first standard region in *SF2* and *SF3* still exceeds the upper limit. This is because the *Dynamic* method is in essential a memory saving optimization technique which may worsen the verification efficiency.

By applying the RRA approach, memory usage are reduced since RRA enables partial loading and verification of grid workflows. Since state spaces of separated regions are also reduced compared to the global workflow, the proposed RRA approach not only reduces verification time, but also the time for BDD operation, Boolean satisfiability solving, memory operations, etc in the implementation of *SMCA* and *BMC* approaches. These result in global performance optimization using the proposed RRA approach.

5. Conclusions

In this work, the decomposition strategy for standard regions based verification of grid workflows is proposed to enhance the performance in formal verification of grid workflow correctness. The RRA approach can effectively decompose a grid workflow into separate standard regions with parallel branches. Consequently, costly global reasoning of a grid workflow can be decomposed into light-weight local reasoning of its standard regions, each with a reduced state space.

The proposed approach is implemented in our *GridPiAnalyzer* automatic formal verification system. Detailed experimental results show that by applying our RRA approach both CPU time and peak memory usage can be dramatically reduced compared to using various traditional formal verification algorithms. Ongoing work includes refinement of the *GridPiAnalyzer* system, implementation of more grid workflow verification modules, and applying the RRA approach to more real world applications.

Acknowledgement

This work is supported by Ministry of Science and Technology of China under the national 863 high-tech R&D program (grant No. 2006AA10Z237) and National Science Foundation of China (grant No. 60604033).

Junwei Cao would like to express his gratitude to Professor Erotokritos Katsavounidis of LIGO (Laser Interferometer Gravitational-wave Observatory) Laboratory at Massachusetts Institute of Technology for his long-term collaboration supports.

References

[1]. D. A. Brown, P. R. Brady, A. Dietz, J. Cao, B. Johnson and J. McNabb, "A Case Study on the Use of Workflow Technologies for Scientific Analysis: Gravitational Wave Data Analysis", in I. J. Taylor, D. Gannon, E. Deelman and M. S. Shields (Eds.), *Workflows for e-Science: Scientific Workflows for Grids*, Springer Verlag, pp. 39-59, 2007.

[2]. J. Cao, S. A. Jarvis, S. Saini and G. R. Nudd, "GridFlow: Workflow Management for Grid Computing", in *Proc. 3rd IEEE/ACM Int. Symp. on Cluster Computing and the Grid*, Tokyo, Japan, pp. 198-205, 2003.

[3]. A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani and A. Tacchella, "NuSMV2: an Open Source Tool for Symbolic Model Checking", *Computer Aided Verification,* LNCS Vol. 2404, Springer Verlag, pp. 359-364, 2002.

[4]. E. M. Clarke, O. Grumberg and D. A. Peled, *Model Checking*, MIT Press, 1999.

[5]. E. Deelman, C. Kesselman, G. Mehta, L. Meshkat, L. Pearlman, K. Blackburn, P. Ehrens, A. Lazzarini, R. Williams and S. Koranda, "GriPhyN and LIGO, Building a Virtual Data Grid for Gravitational Wave Scientists", in *Proc. 11th IEEE Int. Symp. on High Performance Distributed Computing*, Edinburgh, Scotland, pp. 225-234, 2002.

[6]. I. Foster and C. Kesselman, *The GRID: Blueprint for a New Computing Infrastructure*, Morgan-Kaufmann, 1998.

[7]. O. Grumberg and D. E. Long, "Model Checking and Modular Verification", *ACM Trans. on Programming Languages and Systems,* Vol. 16, No. 3, pp. 843-871, 1994.

[8]. R. Milner, *Communicating and Mobile Systems: the Pi Calculus*, Cambridge University Press, 1999.

[9]. Z. Nemeth and V. Sunderam, "Characterizing Grids: Attributes, Definitions, and Formalisms", *J. Grid Computing,* Vol. 1, No. 1, pp. 9-23, 2003.

[10]. G. Salaun, L. Bordeaux and M. Schaerf, "Describing and Reasoning on Web Services Using Process Algebra", in *Proc. IEEE Int. Conf. on Web Services*, San Diego, USA, pp. 43-50, 2004.

[11]. S. Wang and M. P. Armstrong, "A Quadtree Approach to Domain Decomposition for Spatial Interpolation in Grid Computing Environments", *Parallel Computing*, Vol. 29, No. 10, pp. 1481-1504, 2003.

[12]. K. Xu, Y. Wang and C. Wu, "Ensuring Secure and Robust Grid Applications - From a Formal Method Point of View", *Advances in Grid and Pervasive Computing*, LNCS Vol. 3947, Springer Verlag, pp. 537-546, 2006.

[13]. K. Xu, Y. Wang and C. Wu, "Aspect Oriented Region Analysis for Efficient Equipment Grid Application Reasoning", in *Proc. 5th IEEE Int. Conf. on Grid and Cooperative Computing*, Changsha, China, pp. 28-31, 2006.

[14]. K. Xu, Y. Wang, and C. Wu, "Formal Verification Technique for Grid Service Chain Model and its Application", *Science in China Series F – Information Sciences*, Vol. 50, No. 1, pp. 1-20, 2007.

[15]. J. Yu and R. Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing", *J. Grid Computing*, Vol. 3, No. 3-4, pp. 171-200, 2005.