Cloud Resource Selection based on PLS Method for deploying Optimal Infrastructures for Genomic Analytics Application

Katsunori Miura
Kitami Institute of Technology
Kitami, Hokkaido, Japan 090-8507
k-miura@mail.kitami-it.ac.jp

Courtney Powell
Hokkaido University
Sapporo, Hokkaido, Japan 060-0811
kotoni@ist.hokudai.ac.jp

Masaharu Munetomo
Hokkaido University
Sapporo, Hokkaido, Japan 060-0811
munetomo@iic.hokudai.ac.jp

ABSTRACT

This paper proposes a method for determining infrastructures composed of cloud resources that concurrently meet satisfiability and optimality system requirements, such as computational performance, maximum price payable, and deployment location of a genomic analytics application. The input to the proposed method is a mathematical formula that captures the system requirements given by the user, which is defined in accordance with first-order predicate logic, whereas the output is a set of unit clauses representing infrastructures for deploying the genomic analytics application. In the proposed method, an equivalent transformation algorithm is used to generate valid solutions with respect to system requirements, and a genetic algorithm is used to evaluate the optimality of the solutions.

CCS CONCEPTS

• **Information systems** → **Decision support systems**; • **Computing methodologies** → *Knowledge representation and reasoning*;

KEYWORDS

Cloud service, Cloud resource selection, Equivalent transformation, Predicate logic

1 INTRODUCTION

Various methods [2, 3, 5] for finding cloud resources suitable for system requirements from a service catalogue have been studied. This paper proposes a method that determines infrastructures composed of cloud resources that concurrently meet satisfiability and optimality requirements for deploying genomic analytics applications that are specified as workflows comprising multiple analytics tools. The input to the proposed method is a mathematical formula representing the system requirements given by the user, which is defined according to first-order predicate logic, called *Predicate Logic-defined Specification* (PLS).

In a previous study [4], it was shown that the proposed method guarantees the satisfiability of obtained solutions with respect to system requirements, but not their optimality. This paper expands the method in [4] by introducing an evaluation phase in which optimal solutions are generated from the set of valid solutions.

In the proposed method, a computation for cloud resource selection is a series of transformations of clause sets. The initial state of the transformation is a singleton of a definite clause. A clause set is transformed based on an equivalent transformation algorithm (ETA) for guaranteeing that the cloud resources selected by the method are valid with respect to system requirements. In transformations of clause sets by ETA, an answer is described by a unit clause, while the state under computation is described by a definite clause. If there are multiple solutions as valid infrastructure, multiple unit clauses are obtained. Each time a unit clause is obtained by the transformation, the optimality of the infrastructure described by the unit clause is evaluated by the GA. When an obtained infrastructure is evaluated as being optimal by the GA, a unit clause representing the infrastructure is outputted. Otherwise, the result of the GA is reflected to the state under computation, and clause set transformation for determining infrastructures is continued. ETA finds all valid system infrastructures that completely satisfy system requirements, but cannot determine the optimal infrastructures in that set of valid infrastructures. Conversely, GA finds optimal infrastructures but cannot determine their validity with respect to system requirements.

SC17, November 2017, Denver, Colorado USA
2017. ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

2 GENOMIC ANALYTICS APPLICATION

The sample genomic analytics application treated in this paper is a workflow composed of the tools TopHat2, HISAT, and StringTie, and is called RNA sequence or ChIP sequence. In addition, we use workflows that are managed by the genomic science community and actually utilized by users.

3 THE PLS METHOD

3.1 Framework of the PLS Method

The PLS method is a problem-solving method that searches for answers by transforming clause sets. The process is as follows:

Step 1: Construct a PLS reflecting system requirements

A PLS takes the form of a conjunction of universally quantified atomic formula (atom), defined as follows:

$$\forall_{\overline{v}}\{E_1 \wedge E_2 \wedge \cdots E_n\}, \tag{1}$$

where $\overline{v}$ is the set of all variables appearing in (1). The system requirement is represented by multiple atoms, each of which describes a constraint condition.

Step 2: Generate a definite clause *cl* from the PLS

A definite clause *cl* made from a PLS has the following form:

$$ans(v_1, v_2, \cdots v_i) \leftarrow E_1, E_2, \cdots E_n, \tag{2}$$

where the body atoms of *cl* are atoms appearing in the PLS, while the head atom is $ans(v_1, v_2, \cdots v_i)$, which is a special atom representing the answer. The term of the *ans* atom is all variables on $\overline{v}$ in (1). The definite clause (2) means that for a ground substitution θ for all variables on $\overline{v}$, if the constraint conditions on the right hand

side are true, $ans(v_1, v_2, \cdots v_i)\theta$ satisfies the system requirements.
Step 3: Transform clause set $\mathbb{D}$

Clause set $\mathbb{D}$ is transformed to another clause set while preserving the declarative meaning of the initial state $\{cl\}$. For any two clause sets S_1 and S_2, if $\mathcal{M}(S_1) = \mathcal{M}(S_2)$ holds, S_1 is equivalent to S_2 with respect to the declarative meaning [1]. The transformation of clause sets proceeds as follows:

(1) Select a definite clause q from clause set $\mathbb{D}(= \{q\} \cup D)$,
(2) Replace q with clause set Q, where $\mathcal{M}(\{q\}) = \mathcal{M}(Q)$,
(3) Make new clause set $\mathbb{D}$ by joining clause sets D and Q.

The termination criterion is satisfied when a unit clause set or an empty set is obtained as clause set $\mathbb{D}$. The empty set means there is no answer that satisfies the constraint conditions. In addition, when a unit clause representing an optimal infrastructure is obtained, the transformation stops.

3.2 Components of a PLS

An atom is composed of one predicate p and zero or more terms t. In this paper, we define four atoms for describing system requirements for genomic analytics applications:

- $Environment(t_{structure}, t_{workflow})$
- $Location(t_{structure}, t_{location})$
- $Cost(t_{structure}, t_{cost})$
- $Policy(t_{structure}, t_{type}, t_{style})$

A term $t_{structure}$ represents an infrastructure for deploying genomic analytics applications, and other terms represent constraint conditions required for the infrastructure. In this paper, a cloud resource as component of an infrastructure is a virtual instance such as AWS EC2 and Microsoft Azure. The meaning of each atom is as follows: *Environment* atom specifies a genomic analytics application to deploy on an infrastructure. *Location* atom specifies the deployment location of cloud resources. *Cost* atom specifies the maximum price payable for maintaining the infrastructure. *Policy* atom specifies the type of cloud service providers and the combination of cloud service providers in the infrastructure.

4 CLOUD RESOURCE SELECTION

4.1 Framework of Cloud Resource Selection

In transformations of clause set $\mathbb{D}$ (Step 3 shown in Section 3.1), it is important that unit clauses that meet satisfiability and optimality conditions with respect to a PLS are preferentially found. In Step 3, clause set $\mathbb{D}$ is transformed according to the following process.
(1) Select a definite clause q from clause set $\mathbb{D}$

A definite clause q is randomly selected from clause set $\mathbb{D}$. (From the viewpoint of efficient computation, the definite clause selection method is important; however, this process is beyond the scope of this paper.) Let $D(= \mathbb{D} - \{q\})$ be a clause set other than the definite clause q.
(2-1) Replace definite clause q with a clause set Q' using ETA

In ETA, definite clause q is replaced with clause set Q' by a rewriting rule called an equivalent transformation rule (ETR) while preserving the declarative meaning of $\{q\}$. Each ETR is applied to the body atoms of definite clause q. The elements of clause set Q' are either definite clauses or unit clauses. A unit clause obtained by an ETR application represents an infrastructure that satisfies the constraint conditions defined by the atoms shown in Section 3.2; the infrastructure is given as a ground substitution for term $t_{structure}$.
(2-2) Evaluate unit clauses in clause set Q' using GA

All unit clauses are selected from clause set Q', then individuals are made based on the unit clauses. An individual with the minimum value of the following condition is defined as an optimal solution:

- Estimated execution time required for genomic analytics.
- Estimated price required to maintain the infrastructure.

The optimality of a unit clause in clause set Q' is evaluated based on the evolved population.
(2-3) Update definite clauses in clause set Q'

An evaluation atom e is made based on the result of process (2-2), the atom e is added to the body part of the definite clause in clause set Q'. In this paper, a definite clause with an atom e is called a qualified definite clause.
(3) Make new clause set $\mathbb{D}$ by joining clause sets D and Q

Let U be a set of unit clauses appearing in clause set Q'. Let Q be a set obtained by joining a unit clause set U and a set of qualified definite clauses.

4.2 Use of the PLS Method

In this paper, we developed a system for finding cloud resources based on the PLS method. The developed system is applicable to deployment of infrastructures for genomic analytics applications actually utilized by users. For example, the system can automatically reason about the computational performance required for genomic analytics including TopHat or HISAT, and find AWS EC2 instances that satisfy the computational performance.

5 CONCLUSIONS

This paper proposed a new method that makes it possible to find infrastructures that concurrently meet system satisfiability and optimality requirements. The proposed method can be extended to deploy other applications in addition to genomic analytics, without modifying the resource selection algorithm, simply by adding new atoms and their applicable ETRs.

ACKNOWLEDGMENTS

This work was supported by CREST, Japan Science and Technology Agency (Grant No. JPMJCR1501).

REFERENCES

[1] Kiyoshi Akama and Nantajeewarawat Ekawit. 2006. Formalization of the Equivalent Transformation Computation Model. *Journal of Advanced Computational Intelligence and Intelligent Informatics* 10, 3 (September 2006), 245–259.
[2] Saurabh Kumar Garg, Steven Versteeg, and Rajkumar Buyya. 2013. A framework for ranking of cloud computing services. *Future Generation Computer Systems* 29, 4 (2013), 1012–1023.
[3] Nikolay Grozev and Rajkumar Buyya. 2014. Inter-Cloud architectures and application brokering: taxonomy and survey. *Software Prac. Experience* 44, 3 (March 2014), 369–390.
[4] K. Miura, T. Ohta, C. Powell, and M. Munetomo. 2016. Intercloud Brokerages based on PLS Method for deploying Infrastructure for Big Data Analytics. In *Proc. of 2016 IEEE International Conference on Big Data*. 2097–2102. Washington DC, USA.
[5] Smitha Sundareswaran, Anna Cinzia Squicciarini, and Dan Lin. 2012. A Brokerage-Based Approach for Cloud Service Selection. In *The 5th IEEE International Conference on Cloud Computing (CLOUD 2012)*. IEEE, 558–565.