# Kent Academic Repository

**Chen, Huankai, Wang, Frank Z. and Helian, Na (2013)** *A Cost-Efficient and Reliable Resource Allocation Model Based on Cellular Automaton Entropy for Cloud Project Scheduling.* International Journal of Advanced Computer Science and Applications, 4 (4). pp. 7-14. ISSN 2156-5570.

# A Cost-Efficient and Reliable Resource Allocation Model Based on Cellular Automaton Entropy for Cloud Project Scheduling

Huankai Chen
Future Computing Group
University of Kent
Canterbury, UK

Frank Wang
Future Computing Group
University of Kent
Canterbury, UK

Na Helian
University of Hertfordshire
Hatfield
Hertfordshire, UK

*Abstract*—**Resource allocation optimization is a typical cloud project scheduling problem: a problem that limits a cloud system's ability to execute and deliver a project as originally planned. The entropy, as a measure of the degree of disorder in a system, is an indicator of a system's tendency to progress out of order and into a chaotic condition, and it can thus serve to measure a cloud system's reliability for project scheduling. In this paper, cellular automaton is used for modeling the complex cloud project scheduling system. Additionally, a method is presented to analysis the reliability of cloud scheduling system by measuring the average resource entropy (ARE). Furthermore, a new cost-efficient and reliable resource allocation (CERRA) model is proposed based on cellular automaton entropy to aid decision maker for planning projects on the cloud. At last, the proposed model is designed using Matlab toolbox and simulated with three basic cloud scheduling algorithm, First Come First Served Algorithm (FCFS), Min-Min Algorithm and Max-Min Algorithm. The simulation results show that the proposed model can lead to achieve a cost-efficient and reliable resource allocation strategy for running projects on the cloud environment.**

*Keywords*—*Resource Allocation; Cloud Project Scheduling; Entropy; Cellular Automaton; Cost-efficiency; Reliability; Complex System; Local Activity; Global Order; Disorder*

## I. INTRODUCTION

In recent years, Cloud computing is emerging as a new paradigm of large-scale distributed computing, which rent computing resources on-demand, bill on a pay-as-you-go basis, and multiplex many users on the same physical infrastructure. These cloud computing environments provide an illusion of infinite computing resources to cloud users so that they can increase or decrease their resource consumption rate according to the demands. At the same time, resources allocation problem under the cloud environment poses a number of challenges.

Researchers who construct resource allocation strategies for scheduling must cope with the world's natural tendency to disorder. In cloud computing, projects are scheduled on a set of cloud resources that are local active (in the sense that each resource was determined to be assigned tasks based on its own state and the state of the environment and its productivity are affected by the amount of tasks that assigned to it), and corporately structured. We want resource local activity to yield coherent global schedule system order. However, widespread experience warns us that modelling and optimizing systems that exhibit both local activity and global order are not easy. The experience that anything that can go wrong will go wrong and at the worst possible moment is summarized informally as "Murphy's Law" [13]. Scheduling systems are not immune to Murphy. In cloud project scheduling system, after an enough power strikes one of the resources, which leads to its productivity reduced or collapsed, the whole system collapsed. In the real world scenario, such resource productivity reduced or collapsed may cause by hardware/software failures, resources CPU overload, resource over- or under-provisioning, or application misbehaviours. Thus, the system is failed to execute and deliver a project as originally scheduled.

At the root of the ubiquity of disordering tendencies is the Second Law of Thermodynamics, "Energy spontaneously tends to flow only from being concentrated in one place to becoming diffused or dispersed and spread out" [14]. In cloud scheduling system, adding resources to a system may overcome the Second Law "spontaneous tendency" and lead to increasing the system's order. However, the way to decide the numbers of resource allocated to the project is critical. Especially when resources are local active, which is the origin of complexity [12], the scheduling system become more complex under cloud environment. In most case, an increase in the number of assigned resources positively impacts the system's efficiency and reliability. However, there is a limit on the number of assigned resources beyond which any increase may have the opposite effect. Allocate resources beyond this limit may lead to disorder/chaotic condition and a disproportional return on investment in terms of local resource productivity, global system efficiency and reliability.

In the literature, a lot of scheduling algorithms were proposed in the past. Braun et al [6] have studied the relative performance of eleven heuristic algorithms for task scheduling such as First Come First Served (FCFS), Min-Min, Max-Min, Genetic Algorithm (GA), etc. They have also provided a simulation basis for researchers to test the algorithms. A family of 14 scheduling heuristics for concurrently executing BoTs in cloud environments are proposed recently [7]. Most of the past works are mainly aim to shorten project's completion time and enhance the system throughput, which are the focus in improving scheduling algorithm itself. Some theoretical scheduling papers address the reliability problem of scheduling

system by analyzing the entropy produced by scheduling algorithm [3] or resource [1]. However, most of those methods treat scheduling problem as a linear programming problem. We argue that such linear programming technique is not suitable for modelling the complex scheduling system which is dynamic and nonlinear. We are not aware of a method that combine the theoretical analysis of scheduling system with using nonlinear modelling, aiming to achieve both cost-efficiency and reliability of resource allocation strategies, and quantitative measuring the relation between local active resource and global system performance all together.

Both efficiency and reliability are the most important factors for planning a project. The reduced efficiency and reliability of the global system is a direct consequence of the disorder caused as a result of the local active resources and the difficulty in managing these resources. Thus, the resulting resource allocation problem is also an entropy-optimization problem: how many resources should be allocated to a project in order to minimize average resource entropy, subject to limited cost budget within the examined time-period. The fundamental claim of this paper is to solve the above cloud resource allocation problem based on Entropy Theory.

Scheduling is an NP-complete problem, the complexity of which increase substantially in the cloud environment. For such class of problems, in order to achieve the optimal solution an effective method for modelling complex system is also needed. A cellular Automata (CA) is a mathematical model for a complex system which evolves in discrete steps [9]. It is suitable for modelling cloud scheduling system which can be described as a massive collection of resources that interact locally with each other [4]. In this paper, we represent the cloud scheduling system behaviour as a cellular automaton, specifically as a one-dimension cellular automata network.

Following the short introduction on the problem, we will begin discussion by recalling in section 2 the detail problem definition and assumptions of resource allocation for cloud project scheduling. In section 3, the paper presents the general concepts of entropy. Cellular Automation will be applied for modelling the complex scheduling system and a resource allocation model based on cellular automaton entropy will be introduced in section 4 and section 5. We will then describe the experiment and present our simulation results in section 6. Section 7 and Section 8 contains some conclusions and possible future research direction.

## II. PROBLEM DEFINITION

Due to the NP-completeness nature of a scheduling problem, the developed approaches try to find optimal resource allocation solution with considering both cost-efficiency and reliability in the cloud environment. In this paper, the proposed model has been developed under a set of assumptions:

- A project consists of a collection of tasks that have no dependency among each other. Each task requires amounts of computing demand that are known before the task is submitted for execution, or at the time it is submitted.

- Project needs to be completed within deadline and cost budget

- A collection of numbers of cloud resources is rented for running the project. Resources provide amounts of computing capacity. In this paper, Computing capacities were expressed in EC2 compute units (ECU) [15], which for experimental purposes were defined as 1 EC2 compute unit = 1,000,000 million of instructions per second. Hourly cost rates for one ECU were expressed in USD and were based on the EC2 pricing mode [15].

- Selections of one or more scheduling strategies are available for planning the project on the cloud.

In static heuristics, the computing demand for each task is known a priori to execution and measured by ECU. Thus, the expected execution time for a task running on a resource can be calculated by dividing task computing demand by resource computing capacity.

The main aim of scheduling strategies is to minimize a project's completion time and cost with renting a number of resources within deadline. In such scheduling system, the resource allocation problem can be defined as follows:

Let Task set $T = t_1, t_2, t_3 \dots t_n$ be the collection of tasks in a project that submitted to execute on the cloud. Each task requires amounts of computing demand $cd_1, cd_2, cd_3 \dots cd_n$, which is measured by ECU.

Let Resources set $R = r_1, r_2, r_3 \dots r_m$ be the set of resources that are rented for scheduling the tasks. Each resource has its computing capacity which is also measured by ECU, $cc_1, cc_2, cc_3 \dots cc_n$.

Resources are defined as different types according to their computing capacity [15], Resources Type set $RT = rt_1, rt_2, rt_3 \dots rt_k$. The Resource Cost Price Rates for different type are $cp_1, cp_2, cp_3 \dots cp_k$.

The project's completion time, Makespan, can be calculated as follows:

$$Makespan = max(CT_{ij}) \qquad (1)$$

$$CT_{ij} = RT_j + ET_{ij}, where\ 1 < i < n, 1 < j < m \qquad (2)$$

Where $CT_{ij}$ refers to completion time of task $i$ executing on resource $j$, $ET_{ij}$ refers to the expected execution time of task $i$ on resource $j$, and $RT_j$ refers to the ready time of a resource $j$ after completing the previously assigned tasks.

The model we proposed is developed to aid decision makers to solve the following problems:

- How many and what type of resources should we rent?

- How should we schedule the tasks on the rented resources?

So we can achieve a cost-efficient and reliable resource allocation strategy for running the project on the cloud within deadline and cost budget.

## III. ENTROPY THEORY

Entropy is an important statistical quantity which measures the disorder degree and the amount of wasted energy in the transformation from one state to another in a system [14]. Although the concept of entropy was originally a thermodynamic construct, it has been adapted in other fields of study, including information theory, production planning, resource management, computer modelling and simulation [1] [2] [3] [5] [11]. We will use this measure to quantify the reliability degree associated with the scheduling system under different resource allocation strategies. First, we introduce this measure in a general content.

Given a dynamic system X of finite mutually exclusive state variable set $S = s_1, s_2, s_3 \dots s_n$ with probabilities $p_1, p_2, p_3 \dots p_n$ respectively, entropy $H(X)$ is defined as:

$$H(X) = -\sum_{i=1}^{n} p_i \times \log p_i \qquad (3)$$

For any two mutually independent dynamic systems $A$ and $B$ with $n$ and $m$ states respectively, the probability $\pi_{ij}$ of the of the simultaneous occurrence of the states $A_i$ and $B_j$ is $p_i q_j$ where $p_i$ is the probability of state $i$ occurring in system $A$, $q_j$ is the probability of state $j$ occurring in system $B$, where $1 \leq i \leq n$ and $1 \leq j \leq m$. Let the sets of states $A_i B_j$ represent another finite system designated by $AB$. It is easy to see that:

$$H(AB) = H(A) + H(B) \qquad (4)$$

Where $H(AB)$, $H(A)$ and $H(B)$ are the corresponding entropies of systems $AB$, $A$ and $B$.

This expression can be easily extended for an arbitrary number of mutually independent finite systems. For a system $M$ consisting of $s$ mutually independent sub-systems $N_1, N_2, N_3 \dots N_k$, the entropy is given by:

$$H(M) = \sum_{i=1}^{k} H(N_i) \qquad (5)$$

And the average sub-system Entropy [11] is easily obtained by:

$$\bar{H} = \frac{H(M)}{k} \qquad (6)$$

Other properties of this entropy measure, such as those for dependent schemes, can be found, for example, in Khinchin paper [16]. For the purpose of our work, we will only consider mutually independent systems.

## IV. CELLULAR AUTOMATON (CA) AND CA ENTROPY

The theory of cellular automata was initiated by John Von Neumann in his seminal work Theory of Self-Reproducing Automata [17]. It can produce complex phenomenon by simple cell and simple rules, which has the ability to model and simulate the complex system. Since the nineteen eighties, as the evolution of computer technology and the progress of science, cellular automaton theory gets in-depth researched and

is widely applied in economic, transportation, physical, chemical, artificial life and other complex systems [9] [10] [11].

A cellular automaton consists of a regular grid of cells, each in one of a finite number of states, such as Black and White. The grid can be in any finite number of dimensions. For each cell, a set of cells called its neighbour (usually including the cell itself) is defined relative to the specified cell. An initial state (time t=0) is selected by assigning a state for each cell. A new generation is created according to some fixed rules that determine the new state of each cell in terms of the current state of the cell and the states of the cells in its neighbour.

In this work, we model the cloud scheduling system's behaviour as a cellular automaton (CA), specifically as a one-dimension CA network, and then calculate the CA entropy to measure the reliability degree of such complex system under different scheduling rules and resource allocation strategies. In this way, the collection of cells that composes the CA consists of a number of cloud resources that are rented for running the project (Each cell of CA corresponding to a cloud resource). The CA rules in our work are described as selected scheduling algorithms as follows:

- **First Come, First Served (FCFS)**: Tasks are executed according to the sequence of task submitting. The **first come** task will be scheduled on the available resource **first** as soon as it is submitted and then removed from the queue.

- **Min-Min**: All the tasks in a project will be ordered by their computing demands first. The task with the **minimum** computing demand will be scheduled first on the available resource which the completion time is **minimum** and then removed from the queue.

- **Max-Min**: All the tasks in a project will be ordered by their computing demands first. The task with the **maximum** computing demand will be scheduled first on the available resource which the completion time is **minimum** and then removed from the queue.

Each resource gets two performance states: Low Productivity ($LP$) and High Productivity ($HP$), which are correspondingly showed as Black and White in a CA grid map. The state of a resource is determined by its performance ratio under specify scheduling rules. The performance ratio of a resource (RPR) is calculated as follow:

$$RPR = \frac{The\ completion\ time\ for\ all\ its\ assigned\ task}{The\ completion\ time\ of\ the\ project\ (Makespan)} \qquad (7)$$

If the RPR of a resource is over 50%, then it is in High Productivity state, otherwise it is in Low Productivity state.

Reliability is one of the basic characteristics of complex system, which changes with system evolution. For cloud scheduling system, as one resource of it suffered enough power (Such power may cause by internal local activities or external force) strikes, it will fall into low productivity state or at the worst case it breaks down, this is called the resource collapse.

The collapse resource will influence the productivity state of all other resources and may cause them collapse as well,

which lead the scheduling system progress out of order and into a disorder/chaos condition. Along with the increase in the number of collapse resources, hierarchical expansion, will eventually lead to the collapse of the whole scheduling system. Thus, the scheduling system is failed to deliver the project as original planned. We conclude that:

If a system is in order condition, is more reliable, or vice versa. The reliability can be measured by the disorder degree, thus Average Resource Entropy (ARE), of a system.

To evaluate the reliability of scheduling system in CA, we decrease the computing capacity of one resource by 1% for each time step, with a total of 100 time step, which simulates a resource from full computing capacity till break down. The whole scheduling system's evolution pattern is generated and represented by CA grids. Fig. 1 shows some examples of grid pattern generated by CA for running a project consists of 100 random tasks by FCFS algorithm with different number of allocated resources.
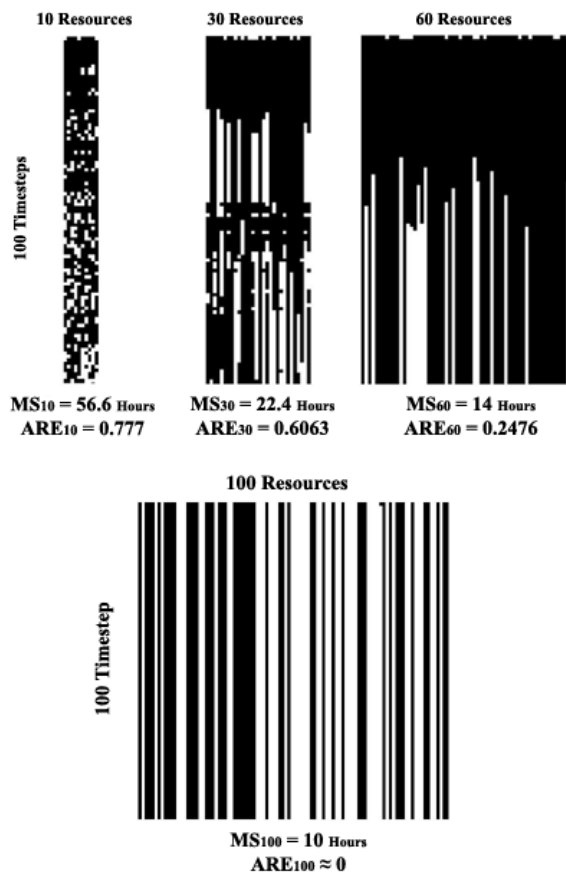


Fig. 1.   Examples of Grid Pattern Generated by Cellular Automaton

The Average Resource Entropy in CA can be calculated by:

$$ARE = \sum_1^n (-p_{LP} \times \log p_{LP} - p_{HP} \times \log p_{HP})/n \qquad (8)$$

Where $n$ refers to the number of resources that rented for running the project, $p_{LP}$ and $p_{HP}$ refers to probability of Low Productivity State and High Productivity State for a resource respectively.

## V.   COST-EFFICIENT AND RELIABLE RESOURCE ALLOCATION (CERRA) MODEL

In this section, a Cost-Efficient and Reliable Resource Allocation (CERRA) model for scheduling project on the cloud is proposed based on the CA Entropy.

The proposed model can be used to achieve the optimal resource allocation strategy by considering both cost-efficiency and reliability for running project on the cloud within deadline and cost budget. The main components and control flow of CERRA model are shown in Fig. 2.
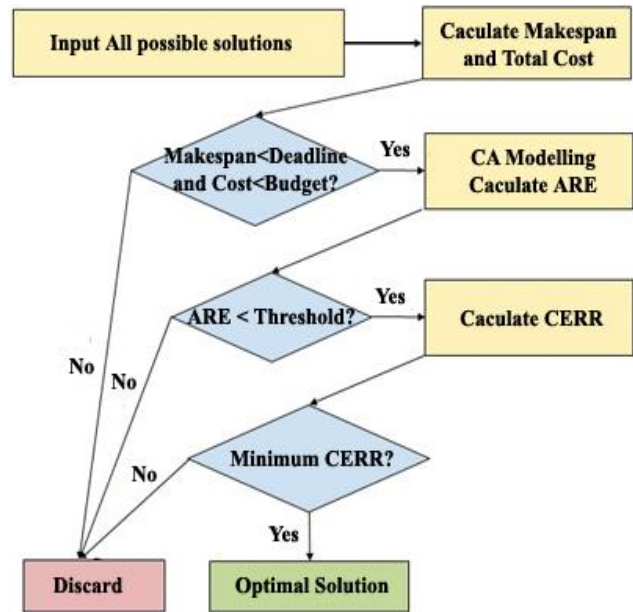


Fig. 2.   Flow Diagram of CERRA Model

The optimal resource allocation solution selected by CERRA model meets the following condition:

- Meeting project deadline and within cost budget

- Under the reliability threshold that user prefer

- With the minimum Cost-Efficiency and Reliability Rate (CERR)

Where the Cost-Efficiency and Reliability Rate (CERR) is calculated by Formula:

$$CERR = \frac{MS_n \times \sum_{i=1}^n cp_i + ARE_n \times (MS_{n-1} - MS_n) \times \sum_{i=1}^n cp_i}{Cost\ Budget} \qquad (9)$$

Where $n$ refers to the number of rented resources to run the project, $MS$ refers to the project's completed time, $cp$ refers to the cost price of a resource and $ARE$ refers to the Average Resource Entropy.

## VI.   EXPERIMENT AND RESULT

We implemented the proposed CERRA model under Matlab environment and simulated with three basic cloud scheduling algorithm, First Come First Served Algorithm (FCFS), Min-Min Algorithm and Max-Min Algorithm.

## A. User Case 1- Simple Project Consists of 10 Random Tasks

A project consists of 10 tasks with random computing demand is listed in Table I. A maximum of 10 cloud resource units are available to be rent for running the project. The type of cloud resource units is M1 Small Instance which is based on Amazon EC2 instance types [17]. The specification of M1 Small Instance is shown in Table II. And the project requirements are shown in Table III.

TABLE I.     PROJECT TASK SPECIFICATION

| Task Specification | Task ID | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* |
| Computing Demand (Hours of 1 ECU) | 12 | 3 | 7 | 9 | 15 | 24 | 10 | 1 | 2 | 4 |

TABLE II.     CLOUD RESOURCE TYPE SPECIFICATION

| Resource Type | Resource Specification | | |
|---|---|---|---|
| | *Computing Capacity* | *Price* | *Available* |
| M1 Small Instance | 1 ECU | $0.115 / Hour | 10 Units |

TABLE III.     PROJECT REQUIREMENTS

| Project Requirements | | |
|---|---|---|
| *Deadline* | *Cost Budget* | *Average Resource Entropy (ARE) Threshold* |
| Makespan<35 Hours | Resource Cost<$20 | ARE < 0.4 |

The experiment results for evaluating three selected scheduling strategies (FCFS, Min-Min, Max-Min) on all possible resource allocation are shown in Fig. 3, Fig. 4 and Fig. 5.

- **Performance Benchmark:**

In general, the makespan of the three scheduling strategies for the project decrease as more resources are rented. However, over a number of resources, e.g. 4 resources for Max-Min and 6 resources for FCFS, any resources that newly invested do not improve the system's performance. With over renting of 5 resources, the improvement is limited for all the scheduling strategies. General speaking, Max-Min strategy performs better than FCFS and Min-Min for most of the solutions. Solutions with less than renting 4 resources are discarded because of failed to meeting the deadline, except for the solution that was allocated 3 resources with Max-Min scheduling strategy.

- **Cost Benchmark:**

In most case, cost of the project linearly increases as more resources are rented. Except for the solutions under Max-Min strategy, the cost for renting 2, 3and 4 resources is similar. Under the cost budget restriction, most of the solutions with renting more than 6 resources are discarded.

- **Reliability Benchmark:**

In general, adding a resource can improve the reliability of the system for all the three scheduling strategies. The reliability improvements for different scheduling strategies vary a lot. In the case of the number of renting resources equals the number
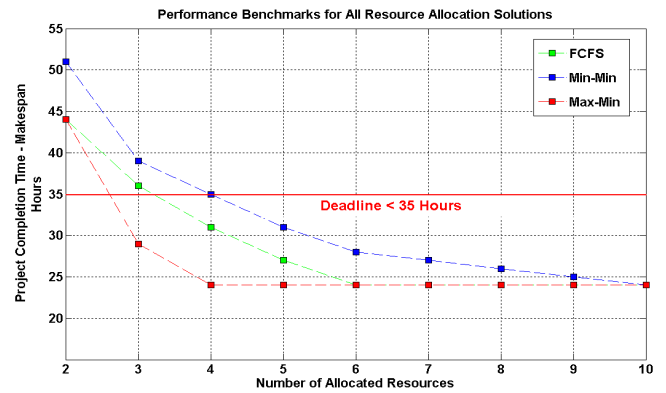


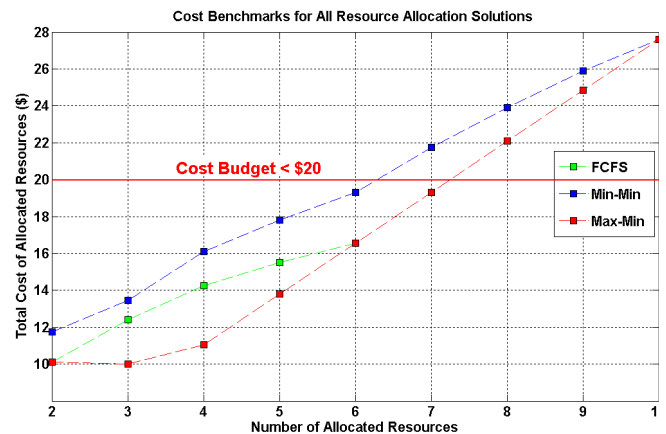Fig. 3.   Performance Benchmarks for All Resource Allocation Solution (10 Tasks)



Fig. 4.   Cost Benchmarks for All Resource Allocation Solution (10 Tasks)
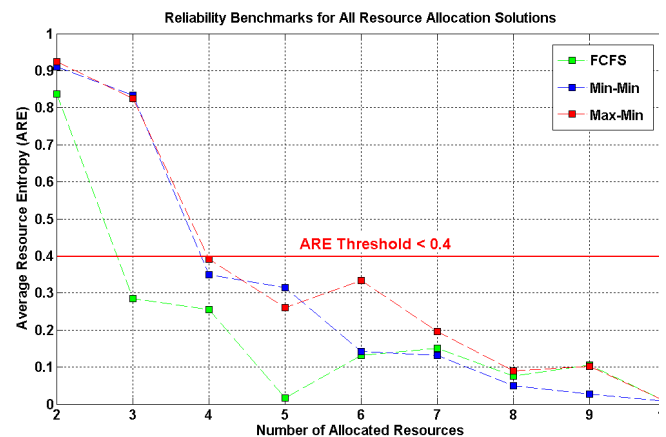


Fig. 5.   ARE Benchmarks for All Resource Allocation Solution (10 Tasks)

of tasks, the project gets as many resources unit as it required and the Average Resource Entropy become zero for all the scheduling strategies.

In this case, scheduling system has zero entropy, indicating order and reliability. For this project, FCFS wins the reliability benchmark in most situations. Most of the solutions with renting less than 4 resources over ARE threshold are discarded.

At last, we calculate the Cost-Efficiency and Reliability Rate for all the resource allocation solutions; the CERR benchmark is shown in Fig. 6. We compare the CERR for all the remaining solutions that meet the project requirements as listed in Table III. With the Minimum CERR principle, the final result and detail performance of the optimal solution are shown in Table IV.
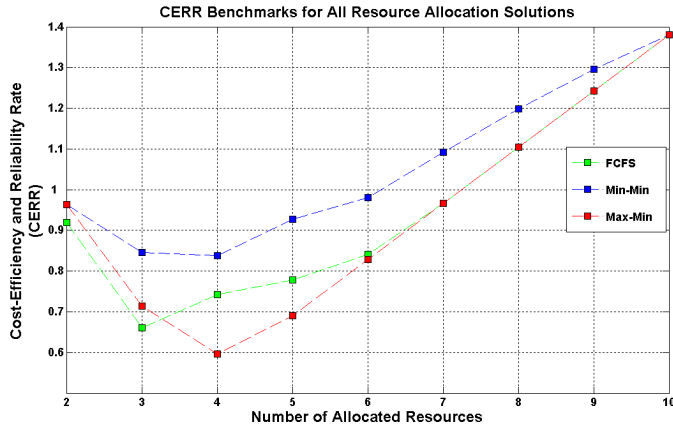


Fig. 6.   CERR Benchmarks for All Resource Allocation Solution (10 Tasks)

As can be seen from the Table IV, solutions with allocating 4 resources for this project are optimal for three of the scheduling strategies. In most case, Max-Min scheduling strategy best fits the project with considering both cost-efficiency and reliability. However, decision maker that prefer to more reliable solution may choose FCFS scheduling strategy as it is more reliable than Max-Min.

TABLE IV.        OPTIMIZE RESOURCE ALLOCATION SOLUTIONS (10 TASKS)

| Solution Specification | Solutions Ranking | | |
|---|---|---|---|
| | First Choice | Second Choice | Third Choice |
| Scheduling Strategy | Max-Min | FCFS | Min-Min |
| Rented Resources | 4 | 4 | 4 |
| Makespan | 24 Hours | 31 Hours | 35 Hours |
| Cost | $11.4 | $14.26 | $16.1 |
| Reliability (ARE) | 0.3914 | 0.2550 | 0.3492 |
| CERR | 0.597 | 0.742 | 0.837 |

## B.  User Case 2 – Complicate Project Consists of 100 Random Tasks

In order to evaluate the robustness of the proposed CERRA model, a more complex project consists of 100 random tasks is presented as shown in Table V. The type of cloud resources and the project requirements are listed in Table VI and Table VII. As the project becomes more complicate, it becomes harder for a decision maker to seek out an optimal solution and make the project manageable. In this case, the reliability of scheduling system is an important factor that cannot be ignored which is related to the risk of failing to running the project as original planned. Thus, if the decision maker chooses a wrong scheduling strategy or resource allocation solution for a project,

it will lead to dramatically increment of project cost or at the worse case failure of finishing the project within deadline. A suitable modelling and accurate measurement of the reliability of system is needed for planning such complicate and large project.

TABLE V.        PROJECT TASK SPECIFICATION

| Project Task Specification | |
|---|---|
| Total Number of Tasks | 100 |
| Total Computing Demand (Hours of 1 ECU) | 5164 |
| Maximum Computing Demand (Hours of 1 ECU) | 100 |
| Minimum Computing Demand (Hours of 1 ECU) | 1 |
| Probability distributions used for creating randomly generated tasks | Normal Distribution: Many middle size tasks, and fewer big and small tasks were contained in the project |

TABLE VI.        CLOUD RESOURCE TYPE SPECIFICATION

| Resource Type | Resource Specification | | |
|---|---|---|---|
| | Computing Capacity | Price | Available |
| M1 Small Instance | 1 ECU | $0.115 / Hour | 100 Units |

TABLE VII.        PROJECT REQUIRENMENTS

| Project Requirements | | |
|---|---|---|
| Deadline | Cost Budget | Average Resource Entropy (ARE) Threshold |
| Makespan<200 Hours | Resource Cost<$800 | ARE < 0.4 |

As Fig. 8 shows, the performances of the three scheduling strategies are quite similar under different resource allocation solutions. However, the costs for different scheduling strategies vary a lot as shown on Fig. 9. Max-Min scheduling strategy wins the cost benchmark for most of the resource allocation solutions. It is no doubt that Max-Min is the selected optimal cost-efficient strategy for this project.

From Fig. 10 we can see the reliability of the system under Max-Min strategy acts like random walk as the number of allocated resources increase. At the point of 30 resources are allocated, the reliability of the system is greatly improved. After that point, the average resource entropy (ARE) of the system increases dramatically and reaches its highest peak at the point of 42 resources then fall back to more order state at the point of 45 resources. In overall, the ARE curve oscillate largely and irregularly until reach the point of 60 resources. In most traditional way, such reliability of a scheduling system is hard to be modelled and measured, which result in being ignored by the decision maker, especially for planning large and complicate projects. With our proposed CERRA model, the above problem can be solved by the quantitative measurement of average resource entropy in the system.

Fig. 7 shows the CERR benchmark for all the resource allocation solutions for the project. Table VIII list the comparisons of several near-optimal resource allocation solutions for running the project under the same Max-Min strategy.
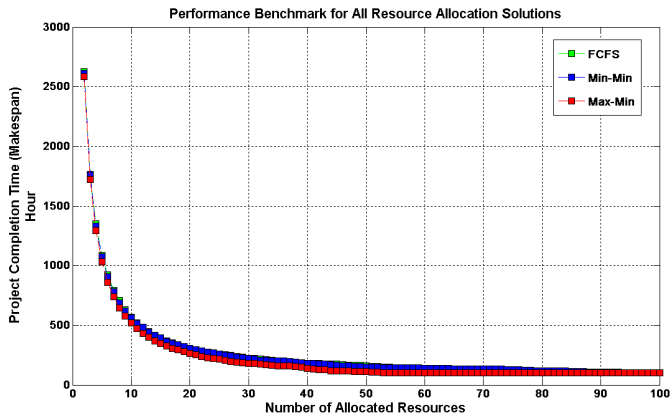
Fig. 8.   Performance Benchmarks for All Resource Allocation Solutions (100 Tasks)
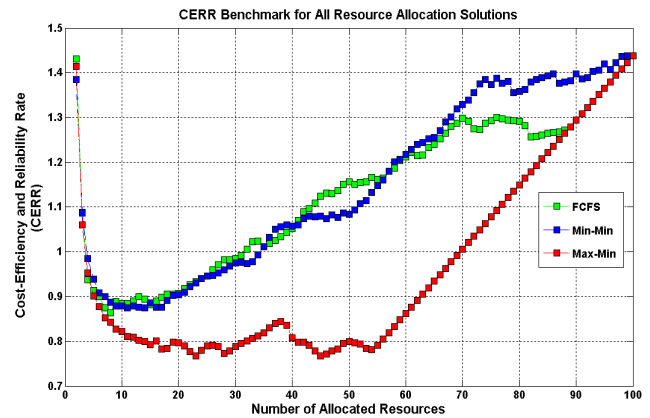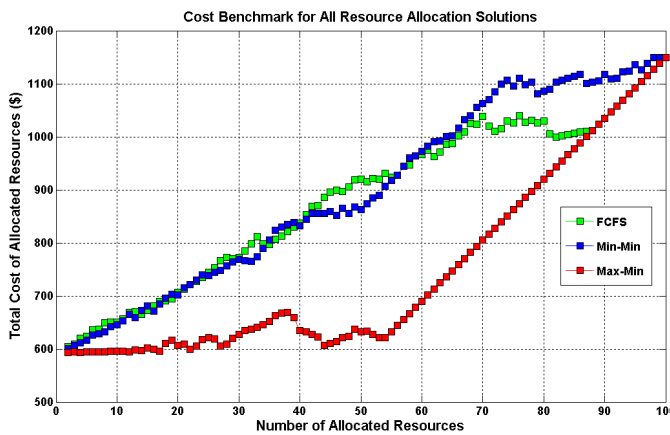


Fig. 9.   Cost Benchmarks for All Resource Allocation Solutions (100 Tasks)



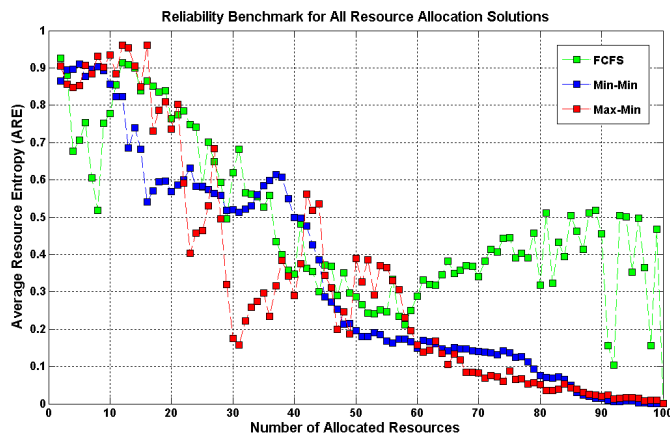Fig. 10. ARE Benchmarks For All Resource Allocation Solutions (100 Tasks)



Fig. 7.   CERR Benchmarks for All Resource Allocation Solution (100 Tasks)

TABLE VIII:     COMPARISON OF DIFFERENT RESOURCE ALLOCATION SOLUTIONS UNDER MAX-MIN SCHEDULING STRATEGY

| Max-Min Scheduling Strategy | Resource Allocation Solutions Comparisons | | | | |
|---|---|---|---|---|---|
| | *1* | *2* | *3* | *4* | *5* |
| Rented Resources | 23 | 30 | 38 | 44 | 45 |
| Makespan (Hour) | 229 | 182 | 153 | 120 | 118 |
| Cost ($) | 605.71 | 627.90 | 668.61 | 607.20 | 610.65 |
| Reliability (ARE) | 0.404 | 0.175 | 0.384 | 0.536 | 0.343 |
| CERR | 0.768 | 0.787 | 0.844 | 0.779 | 0.767 |

Form Table VIII, some observations were drawn.

- Observation 1: With minimum CERR principle, solution 5 is selected as the optimal solution for running the project.

- Observation 2:  Although the solution 4 is discarded because its reliability degree (0.536) is over ARE threshold (ARE<0.4). It is still a near-optimal solution that performs close to solution 5.

- Observation 3: Compare solution 3 with solution 2 and 4, we can see solution with an allocation of 38 resources for the project result in disproportional return on investment.

- Observation 4: The CERR value of solution 1 is close to solution 5 with similar cost and reliability degree but huge performance difference. Since we measure the CERR by considering strictly "meeting the deadline" only, excluding the saving time cost of meeting the deadline. In the future, this factor should be considered in our CERRA model.

In summary, the proposed CERRA model is capable of providing useful information and quantitative measurement for aiding the decision maker to achieve a Cost-Efficient and Reliable solution for planning projects on the cloud.

## VII. CONCLUSION

Resource Allocation in cloud scheduling system is a complex problem, the solution of which requires suitable modelling and complex optimization calculations. The CERRA model proposed in this paper puts forward an optimization method that is different from the traditional approach. It is one that is based on Cellular Automaton Entropy, based on minimizing the CERR of a scheduling system, which indicates both cost-efficiency and higher level of reliability resource allocation solution thus a more manageable project. The proposed model has been applied to aid decision maker for planning project on the cloud. The experiments help demonstrate how the CERRA model can be implemented and interpreted, and how the CA Entropy-based solution can be introduced in a project manager's decision-making process. The experiment result shows that the proposed model is able to achieve both cost-efficient and reliable resource allocation solution for running project on the cloud by solving the questions when planner making a decision:

- How many resources do I need?

- How should I schedule the project on the resources?

- Is it such solution cost-efficient and reliable?

- Giving a collection of solutions, which one is better for different requirements?

## VIII. FUTURE WORK

Since the approach of applying Cellular Automaton Entropy to analysis the cloud scheduling system in this paper is the first attempt in the related literature. Many problems may arise, and many issues remain open. Future work should further examine and expand the entropy method presented in this paper: (1) we would like to determine if our model can be generalized for other types of complex scheduling strategies, e.g. Genetic Algorithm (GA), DAG scheduling algorithm, Simulated Annealing (SA), Ant colony optimization (ACO) Algorithms; (2) We would also like to study the optimization problem when allowing for the heterogeneous types of cloud resources and the dynamic adjustment of the number of cloud resources during run-time; (3) Also we are currently developing our model as a web service to help user in improving their resources renting strategies in the Cloud environment; (3) Finally, we plan to research on its application in other similar real-world problems, e.g. Staff Shifting Management, Calling Centre Scheduling, Traffic Routing, Manufactory Production.

REFERENCES

[1] Christodoulou, Symeon, Georgios Ellinas, and Pooyan Aslani. "Entropy-based scheduling of resource-constrained construction projects." *Automation in Construction* 18.7 (2009): 919-928.

[2] Hermenier, Fabien, et al. "Entropy: a consolidation manager for clusters." Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments. ACM, 2009.

[3] Gan, H-S., and A. Wirth. "Comparing deterministic, robust and online scheduling using entropy." International journal of production research 43.10 (2005): 2113-2134.

[4] Botón-Fernández, María, Francisco Prieto Castrillo, and Miguel Vega-Rodríguez. "Nature-inspired algorithms applied to an efficient and self-adaptive resources selection model for grid applications." Theory and Practice of Natural Computing (2012): 84-96.

[5] Liu, Jiansheng, et al. "Research on Measurement Entropy-Based of Equipment Management Complexity and Its Application in Production Planning." Intelligent Robotics and Applications (2008): 604-611.

[6] Braun, Tracy D., et al. "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems." Journal of Parallel and Distributed computing 61.6 (2001): 810-837.

[7] Gutierrez-Garcia, J. Octavio, and Kwang Mong Sim. "A family of heuristics for agent-based elastic Cloud bag-of-tasks concurrent scheduling." *Future Generation Computer Systems* (2012).

[8] Dong, Fangpeng, and Selim G. Akl. "Scheduling algorithms for grid computing: State of the art and open problems." School of Computing, Queen's University, Kingston, Ontario (2006).

[9] Toffoli, Tommaso, and Norman Margolus. Cellular automata machines: a new environment for modeling. MIT press, 1987.

[10] Wolfram, Stephen. "Universality and complexity in cellular automata." Physica D: Nonlinear Phenomena 10.1 (1984): 1-35.

[11] Langton, Chris G. "Computation at the edge of chaos: Phase transitions and emergent computation." Physica D: Nonlinear Phenomena 42.1 (1990): 12-37.

[12] LEON, O. "Local activity is the origin of complexity." International journal of bifurcation and chaos 15.11 (2005): 3435-3456.

[13] Matthews, Robert AJ. "The science of Murphy's Law." SCIENTIFIC AMERICAN-AMERICAN EDITION- 276 (1997): 88-91.

[14] Lambert, F.L.: The Second Law of Thermodynamics. http://www.secondlaw.com, 2005.

[15] Amazon EC2, http://aws.amazon.com/ec2/, 2013.

[16] Khinchin, A. Ya. Mathematical foundations of information theory. Dover Publications, 1957.

[17] Von Neumann, John, and Arthur W. Burks. "Theory of self-reproducing automata." (1966)