УДК 004.27
DOI: 10.18523/2617-3808.2024.7.25-30

B. Artiushenko

A NUCLEOLUS-BASED APPROACH FOR CLOUD RESOURCE ALLOCATION

Cloud computing has transformed organizational operations by enabling flexible resource allocation and reducing upfront hardware investments. However, the growing complexity of resource management, particularly for computing instances, has led to challenges in cost control and resource allocation. Fair allocation policies, such as max-min fairness and Dominant Resource Fairness, aim to distribute resources fairly among users. In recent years, the FinOps framework has emerged to address cloud cost management, empowering teams to manage their own resource usage and budgets. The allocation of resources among competing product teams within an organization can be modelled as a cooperative game, where teams with competing priorities must negotiate resource allocation based on their claims and the available budget.

The article explores cloud resource allocation as a cooperative game, particularly in situations where the total budget is insufficient to meet all teams' demands. Several resource allocation methods are discussed, including the proportional rule and the nucleolus-based approach, which seeks to minimize the coalitions' incentives to deviate. The nucleolus method offers a stable and fair solution by distributing resources in a way that maximizes stability and reduces the likelihood of coalitions deviating from the overall allocation. This approach ensures that no team is allocated more than its claim and maintains fairness by adhering to principles such as claim boundaries, monotonicity, and resource constraints. Ultimately, the nucleolus-based method is proposed as an effective solution for allocating cloud resources in a cooperative and stable manner, ensuring that resource allocation is both fair and efficient.

Keywords: cloud computing, resource allocation, bankruptcy issue, cooperative game, nucleolus.

Introduction

Cloud computing is a game-changing technology that revolutionized the way many organizations operate. With cloud technologies, organizations can operate more flexibly, easily scaling and reducing up-front investments in hardware. Thus, the cost of cloud computing usage could quickly escalate [3].

Resource allocation is an important part of any shared system, not just clouds. One of the most popular allocation policies proposed so far has been max-min fairness, which maximizes the minimum allocation received by a user in the system. Assuming each user has enough demand, this policy gives them an equal share of resources. Round-robin, proportional resource sharing, and weighted fair queueing are known algorithms of this approach. A shared-system specialized allocation method, like Dominant Resource Fairness, was proposed for multiple resource types [4] and could be used for clouds as well [2].

These methods approach the idea of fairness, where each user is allocated more or less in proportion to their demand and weight of resources. In other words, allocation is viewed as a non-cooperative game in which alliances cannot be formed, or all agreements need to be self-enforcing [1].

In recent years a new profession, 'FinOps practitioners,' emerged, along with a framework to address-cloud cost management. One of the main FinOps principles is that 'individual feature and product teams are empowered to manage their own cloud usage within their budget.' [6]

In other words, each team should take part in computing resource allocations for their needs and managing costs. Whereas some resources (object storage, selective databases) are dynamically allocated, computing instances require provisioning time and are often are allocated in advance.

Usually, resources need to be allocated among product teams. These teams have competing priorities, and each product team is focused on its respective product, not the system as a whole. The teams can form coalitions, and there is a possibility of external enforcement of cooperative behavior (formal or informal). Even though 'common goods' could be facilitated within the organization, factions could easily form and could be considered.

Cloud Resource Allocation as a Cooperative Game

Suppose there are N product teams, each demanding computing resources such as CPU, memory size, and memory bandwidth. Within serverless computing, most resources can scale on demand, while some need to be allocated in advance. Let's consider allocating computing instances in a specific cloud.

Most major cloud vendors provide a selection of instances with various hardware and price options. In a cloud environment, all resources can be summarized by a single number: price (per usage). Table 1 provides an example of pricing for various general-purpose computing instances (EC2) of AWS in the US East (Ohio) retrieved on 2024-04-01. To eliminate cents, let us consider hourly prices multiplied by 168 (weekly), with $/w representing USD per week.

Table 1. **AWS EC2 prices and key performance indicators**

Instance	v CPU	Clock Speed	Memory	EBS Mbps	Price $/w
m5.large	2	3.1 GHz	8 GiB	<2,120	17
m4.large	2	2.4 GHz	8 GiB	450	18
m5.xlarge	4	3.1 GHz	16 GiB	<2,120	34
m4.xlarge	4	2.4 GHz	16 GiB	750	37
m5.2xlarge	8	3.1 GHz	32 GiB	<2,120	68
m4.2xlarge	8	2.4 GHz	32 GiB	1,000	74
m5.4xlarge	16	3.1 GHz	64 GiB	2,120	137
m4.4xlarge	16	2.4 GHz	64 GiB	2,000	148
m5.8xlarge	32	3.1 GHz	128 GiB	5,000	274
m4.10xlarge	40	2.4 GHz	160 GiB	4,000	336
m5.12xlarge	48	3.1 GHz	192 GiB	7,000	411
m4.16xlarge	64	2.4 GHz	256 GiB	10,000	538
m5.16xlarge	64	3.1 GHz	256 GiB	10,000	548
m5.24xlarge	96	3.1 GHz	384 GiB	12,000	774

Each product team can select the main characteristics of instances and choose desired instances based on the available list of computing instances and actual needs. Let us assume all product teams use the same utilization factor and other metrics to make their decision. For example, there are three teams (N=3) with specific requests (see Table 2), and allocation is on a weekly basis. Then, based on each team's request, FinOps (assuming it is responsible for resource allocation) estimates the total budget. For simplicity, let us assume it is also calculated on a weekly basis.

The task is: how should we allocate resources when the total requested budget (total demand) is higher than the available budget (estate)? Obviously, if the estate is greater than the total demand, it is not a problem.

Table 2. **Product teams' desired instances**

Team	Main characteristic	Asked instance	Ask, $/w
1	Memory	m5.8xlarge	274
2	EBS	m4.16xlarge	538
3	CPU	m5.24xlarge	774
Total			1,586

For example, let the estate be $600/w, whereas the total demand (see Table 2) is $1,586/w. Please notice that Team 3's request ($774/w) alone is higher than the available budget ($600/w).

This task can be rewritten as a claims problem (also called a bankruptcy problem). For that, the budget should be considered as a divisible source. Each product team is a player, and each team's desired instance price can be considered its claim or request. If all product teams use the same metrics and have the same value, we can consider a claim to be a team's weight as well. The available budget is the estate. Deficits are the differences between a team's claims and awards.

Is cloud resource allocation between product teams an example of a cooperative game? A cooperative game is one where players are able to make enforceable contracts outside of those specifically modelled within the game. FinOps should find a solution that minimizes the chances of any team 'going upstairs' or teams merging/splitting just to cheat the system. In other words, no single team or coalition will leave the grand coalition (all teams combined). Thus, the problem can be considered a cooperative game. Excess is

the difference between the payment given to the coalition for its allocated resources and the value the coalition would receive by deviating.

What are the properties of a good solution?

1. Claim boundaries: No team gets more than its claim or less than $0.
2. Equal treatment of equals: If two teams have the same claim, they should be given (roughly) equal awards.
3. Monotonicity: Teams with higher original claims should not receive fewer awards.
4. Resource monotonicity: If the estate increases, each team should not get less.
5. Cloud constraints: A finite number of available resource types and possible awards.

Maximum Award

Let us start with the goal of finding a solution that maximizes the total award alone. The solution can be found by brute force (see Table 3. Hereafter, a coalition is described by a tuple of teams, for example, (1,2,3) represents a grand coalition for all three teams. When all coalitions have only positive excesses, then deviation is not profitable for any coalition.

How is excess calculated? Let us take coalition (2,3) as an example. If it deviates, then the total amount allocatable to it will be $600/w minus $538/w (claim of the remaining coalition (1)) giving $62/w to allocate. What could this coalition buy with the allocated funds? The best option is an m4.large ($18/w) and an m4.xlarge ($37/w) with a total award of $55/w. Thus, the excess is its award of $582/w minus $55/w, resulting in $527/w.

Thus, monotonicity is not maintained, as the award for coalition (2) is less than the award of coalition (1), so additional conditions must be added.

Table 3. **Maximum award solution**

Coalition	Award, $/w	Instance(s)	Excess, $/w
(1)	34	[m5.xlarge]	34
(2)	18	[m4.large]	18
(3)	548	[m5.16xlarge]	548
(1, 2)	52	[m5.xlarge, m4.large]	52
(1, 3)	582	[m5.xlarge, m5.16xlarge]	527
(2, 3)	566	[m4.large, m5.16xlarge]	255
(1, 2, 3)	600	[m5.xlarge, m4.large, m5.16xlarge]	

Constraint Equal Award Rule

One common approach is to distribute available resources proportionally to the estate, divided equally among all teams (without exceeding their claims). An example of the rule's application is given in Table 4.

Table 4. **Constraint equal award rule allocation**

Coalition	Award, $/w	Instance(s)	Excess, $/w
(1)	148	[m4.4xlarge]	148
(2)	148	[m4.4xlarge]	148
(3)	148	[m4.4xlarge]	148
(1, 2)	296	[m4.4xlarge, m4.4xlarge]	296
(1, 3)	296	[m4.4xlarge, m4.4xlarge]	241
(2, 3)	296	[m4.4xlarge, m4.4xlarge]	-15
(1, 2, 3)	444	[m4.4xlarge, m4.4xlarge, m4.4xlarge]	

As seen in Table 4, the excess of coalition (2,3) is negative. How is it possible? Suppose that coalition deviates from the grand coalition. The claim of the remaining coalition (1) is $274/w. It can be fully granted, as the m5.8xlarge instance price is exactly $274/w. The remaining budget is $600 - $274 = $326/w. We can allocate the total budget by allocating an m5.8xlarge ($274/w) to Team 2 and an m4.xlarge ($37/w) to Team 3. This means that coalition (2,3) will receive $311/w, which is higher than the allocated $296/w, making deviation from the grand coalition beneficial. Thus, the equal award is not a stable solution, as for some of the product team coalitions, deviation is more beneficial.

Proportional Rule

Many authors [2] consider the proportional rule a reasonably fair allocation method. In this approach, each player receives a share of the estate proportional to its weight. An example of the rule is given in Table 5. Obviously, for product team (3), this approach is much better than the previous one. Please note, that for the coalition of teams (2,3) it is more profitable to deviate as excess is negative. Even so, competing coalition (1) receives less, making this solution appear to be more stable.

Table 5. **Proportional rule**

Coalition	Award, $/w	Instance(s)	Excess, $/w
(1)	74	[m4.2xlarge]	74
(2)	148	[m4.4xlarge]	148
(3)	148	[m4.4xlarge]	148
(1, 2)	222	[m4.2xlarge, m4.4xlarge]	222
(1, 3)	222	[m4.2xlarge, m4.4xlarge]	167
(2, 3)	296	[m4.4xlarge, m4.4xlarge]	-15
(1, 2, 3)	370	[m4.2xlarge, m4.4xlarge, m4.4xlarge]	

Nucleolus for Computing Cloud Resource Allocation

The nucleolus is a solution concept in cooperative game theory that maximizes stability by minimizing coalitions' incentives to deviate [5].

The objective is to find an imputation whose excesses vector is maximized in leximin order within the available range of choices per player.

Table 6 shows that the nucleolus solution provides both the best excess and deficit values for the worst coalitions, making it the most stable choice.

Table 6. **Nucleolus**

Coalition	Award, $/w	Instance(s)	Excess, $/w
(1)	148	[m4.4xlarge]	148
(2)	148	[m4.4xlarge]	148
(3)	274	[m5.8xlarge]	274
(1, 2)	296	[m4.4xlarge, m4.4xlarge]	296
(1, 3)	422	[m4.4xlarge, m5.8xlarge]	367
(2, 3)	422	[m4.4xlarge, m5.8xlarge]	111
(1, 2, 3)	570	[m4.4xlarge, m4.4xlarge, m5.8xlarge]	570

Thus, the nucleolus should be determined within a limited choice space, after which FinOps can calibrate resource allocation based on each product team's main computing characteristics.

The suggested approach is as follows:

1. Define claims (instance price) and budget in game theory terms.
2. Find a solution that maximizes the smallest excess of a coalition (nucleolus) in relation to available cloud resources.
3. For each product team select an instance with a price lower than the allocated profit based on the product team's desired criteria.

The suggested approach is presented as a BPMN model in Figure 1.

Found nucleolus per each estate and possible instance types taken from Table 1 are shown in Table 7. One may notice that claim boundaries, monotonicity, resource monotonicity, and cloud constraints are maintained in the given examples.

Table 7. **Nucleolus-based solutions for different estates**

Estate, $/w	Awards, $/w		
	Product team 1	Product team 2	Product team 3
222	74	74	74
370	74	148	148
570	148	148	274
758	148	274	336

Estate, $/w	Awards, $/w		
	Product team 1	Product team 2	Product team 3
1107	148	411	548
1586	274	538	774

Based on the identified nucleolus-based solution, appropriate instances can be assigned (see Table 8).

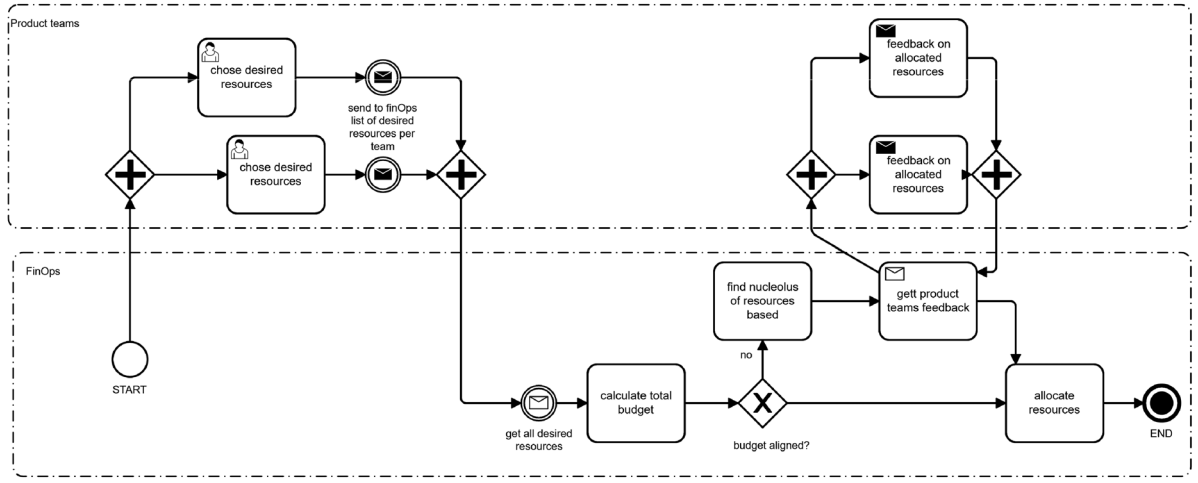

Figure 1. Suggested approach

Table 8. **Allocated instances**

Estate, $/w	Product team 1	Product team 2	Product team 3
222	m4.2xlarge	m4.2xlarge	m4.2xlarge
370	m4.2xlarge	m4.4xlarge	m4.4xlarge
570	m4.4xlarge	m4.4xlarge	m5.8xlarge
758	m4.4xlarge	m5.8xlarge	m4.10xlarge
1107	m4.4xlarge	m5.12xlarge	m5.16xlarge
1586	m5.8xlarge	m4.16xlarge	m5.24xlarge

Thus, the m4.4xlarge has lower EBS than the m5.4xlarge, and product team 2's priority is EBS. The same applies to product team 3: CPU is their highest priority. Based on these requirements, some product teams may be offered an alternative solution (see Table 9), as their main characteristic requirements sometimes prove to be more flexible at later stages.

Table 9. **Allocated instances based on total budgets and usage**

Estate, $/w	Product team 1	Product team 2	Product team 3
222	m4.2xlarge	m4.2xlarge	m5.2xlarge
370	m4.2xlarge	m4.4xlarge	m4.4xlarge
570	m4.4xlarge	m4.4xlarge	m5.8xlarge
758	m4.4xlarge	m5.2xlarge	m5.8xlarge
1107	m4.4xlarge	m5.12xlarge	m5.16xlarge
1586	m5.8xlarge	m4.16xlarge	m5.24xlarge

Conclusions

A game-theoretic approach to allocating cloud-based computing resources has been presented. In the computational service (or resource) market, each service incurs a cost based on the amount of resources used. Specifically, the allocation of cloud computing instances among product teams is examined. As each resource is allocated to a specific team and teams can cooperate with one another, the task can be considered a cooperative game.

A nucleolus-based approach is suggested to solve the cloud resource allocation problem. First, a nucleolus is determined based on the available cloud resource options, and then the solution is adjusted based on the product teams' objectives. Using the nucleolus maximizes the stability of resource allocation by minimizing the incentives of product team coalitions to deviate.

Список літератури

1. A game-theoretic method of fair resource allocation for cloud computing services / G. Wei et al. // The journal of supercomputing. — 2010. — Vol. 54. — Pp. 252–269. — https://doi.org/10.1007/s11227-009-0318-1.
2. Bei X. Fair and efficient multi-resource allocation for cloud computing / X. Bei, Z. Li, J. Luo // Web and internet economics 18th international conference. — Troy, NY, 2022. — Pp. 169–186.
3. Deochake S. Cloud cost optimization: a comprehensive review of strategies and case studies / S. Deochake. — 2023. — 36 p. (Preprint). — https://doi.org/10.48550/arXiv.2307.12479.
4. Dominant resource fairness: fair allocation of multiple resource types [Electronic resource] / A. Ghodsi et al. // 8th USENIX symposium on networked systems design and implementation. — Boston, MA, 2011. — Mode of access: https://www.usenix.org/conference/nsdi11/dominant-resource-fairness-fair-allocation-multiple-resource-types. — Title from screen.
5. Schmeidler D. The nucleolus of a characteristic function game / D.Schmeidler // SIAM journal on applied mathematics. — 1969. — Vol. 17, no. 6. — Pp. 1163–1170. — https://doi.org/10.1137/0117107.
6. What is FinOps? [Electronic resource] // FinOps Foundation. — Mode of access: https://www.finops.org/framework/ (date of access: 28.12.2024). — Title from screen.

References

Bei, X., Li, Z., & Luo, J. (2022). Fair and efficient multi-resource allocation for cloud computing. *Web and internet economics 18th international conference* (pp. 169–186). Springer.

Deochake, S. (2023). *Cloud cost optimization: a comprehensive review of strategies and case studies*. https://doi.org/10.48550/arXiv.2307.12479.

Ghodsi, A., Zaharia, M., Hindman, B., Konwinski, A., Shenker, S., & Stoica, I. (2011). Dominant resource fairness: fair allocation of multiple resource types. У *8th USENIX symposium on networked systems design and implementation*. USENIX Association. https://www.usenix.org/conference/nsdi11/dominant-resource-fairness-fair-allocation-multiple-resource-types.

Schmeidler, D. (1969). The nucleolus of a characteristic function game. *SIAM journal on applied mathematics*, *17* (6), 1163–1170. https://doi.org/10.1137/0117107.

Wei, G., Vasilakos, A. V., Zheng, Y., & Xiong, N. (2010). A game-theoretic method of fair resource allocation for cloud computing services. *The journal of supercomputing*, *54*, 252–269. https://doi.org/10.1007/s11227-009-0318-1.

What is FinOps? FinOps Foundation. https://www.finops.org/framework/.

Артюшенко Б. А.

РОЗПОДІЛ ХМАРНИХ РЕСУРСІВ НА БАЗІ НУКЛЕОЛУСА

Хмарні обчислення змінили процес менеджменту інфраструктури та ввели нові виклики, зокрема керування витратами на хмару. В роботі розглянуто розподіл хмарних ресурсів із метою оптимізації витрат. Показано, що розподіл хмарних ресурсів можливо розглядати як приклад кооперативної гри. Запропоновано підхід на базі нуклеолуса для розв'язання задачі максимізації найгіршого ексцесу коаліції при розподілі хмарних ресурсів. Розглянуто й досліджено приклади наближені до реальних із порівнянням з поширеними підходами.

Ключові слова: хмарні технології, розподіл ресурсів, кооперативна гра, нуклеолус.

Матеріал надійшов 14.03.2024