The Serverless Computing for Acceptance and Shipping Warehouse Zones Optimization

Oleksandr Muliarevych

Lviv Polytechnic National University, Stepana Bandery St. 14, Lviv, 79000, Ukraine

Abstract

The article is devoted to a key part of any e-commerce – customer's order execution. After overview of the main warehouse operations the input parameters and their potential combinations of acceptance and shipping zones calculation are described, which is a part of warehouse design complex task. This study demonstrates complexity of searching for optimal design solution because of big number of input parameters and constraints. In worst case, hundred of thousands parameter combinations should be evaluated to find the optimal one. Developed warehouse design system based on prediction model and serverless computing, which evaluates different parameter combinations decreases time for optimal result search and makes warehouse design more automated. Achieved results demonstrated in article help to compare efficience of different approaches for the most time-consuming part – computing sub-system: monolith architecture, microservices with horizontal auto-scaling and serverless light-weighted functional processing.

Keywords [1]

Warehouse design, serverless, computing, acceptance and shipping zones, labor cost

1. Introduction

Nowadays, online shops are spread widely, everyone can search using internet any goods and order them in several clicks. Under the hood, there are lot of business processes should be well organised to make this possible, solving issues with the storage space rent and goods flows transfer and routing. Warehouse operation is one of the main problems that appears for different industries and business models that are not conducted with building and design new warehouse buildings [1]. This fact causes a high popularity of such called "logistics outsourcing" or when between 2 typical participants in a goods flow: supplier and customer appear the third, who helps with goods storage, distribution, and transfer, it is called also "third-party logistic" provider (3PL) [2]. Third-party logistics provider – is usually a firm which provides multiple logistics services for use by customers related to facilitate the movement of parts and materials from suppliers to manufacturers and finished products from manufacturers to distributors and retailers. Goods transportation, warehousing, cross-docking, inventory management, packaging and freight forwarding are provided together as a package of services by the 3PL provider [3].

Using 3PL providers, customers achieve next benefits: better quality of services, cost reduction, better and easier control, focus on core competences, better order accuracy and time compression. Customers doesn't change warehouses often, usually it's a long-term contract, which brings profit for both side: manufacturer who rent a part of storage capacities with full set of order processing services and 3PL provider who manage warehouse and try to keep utilization level of building at max ratio, win-win strategy. The problem appears at the start point when customer has estimated input order data required to be processed during next year, and he is searching for the best option with a lowest cost and fast response between all available in his region 3PL providers. The same situation repeats when goods volume changes and rebalance of estimated zone should be done as fast as possible. For sure, it's a

Information Technology and Implementation (IT&I-2022), November 30 - December 02, 2022, Kyiv, Ukraine
EMAIL: oleksandr.v.muliarevych@lpnu.ua (A. 1);
ORCID: 0000-0002-4644-7962 (A. 1);

usual task for professional logistics engineers [4], but a lot of manual calculation has several disadvantages: 1) human failure probability on some step that is hard to detect; 2) in the case of huge portions of input order data and big list of constraints from a customer, it could increase calculation complexity and time for final result definition till the month or even more [2]; 3) set of options that would be checked by professionals doesn't guarantee that the best optimal by cost design option wouldn't be missed, because full combinatorial analysis is impossible task for manual execution.

That's why developing software and computer systems for logistics, especially automated warehouse design, is so popular nowadays [3] and Deopware.com platform that includes methodology and approaches described in this article isn't an exclusion.

2. Warehouse Design

Mentioned before Deopware.com platform as a software platform suite for computing and evaluation of warehouse design efficiency, it offers reliable, scalable, and flexible services based on mathematical algorithms and the latest IT solutions – combining pattern recognition [5] and learnable prediction model [6], serverless computing [7] and microservice architecture [8].

Firstly, let's overview what type of zones warehouse consists of and what list of parameters, constraints and input order data used for evaluation. The processing areas for the goods flow are next [4]: 1) unloading and acceptance zone; 2) storage and collection zone; 3) control and picking zone; 4) transport expedition zone; 5) shipping zone. This typical zoning is shown in Figure 1., where four rectangle-zones inside of an abstract warehouse building are represented.

Usually input data for design warehouse task contains next data: 1) warehouse operation mode (expected working hours and time shifts); 2) delivery standard (frequency and volume of income supplies, income pallets, packaging units properties; expected income vehicles parameters); 3) storage standard (extrapolated and adjusted set of orders and products, using which we can evaluate the volume of commodity flow and it's unevenness ratio; set of racks, additional criterias for a forward-reserve problem solving [9]); 4) selection and shipping standard (order packaging units, outbound vehicles parameters, shipment working hours, due dates for delivery); 5) transporters standard (warehouse specific transposters properties); 6) optional building constraints (size of building, piles, forbidden areas, existed communications, etc.).

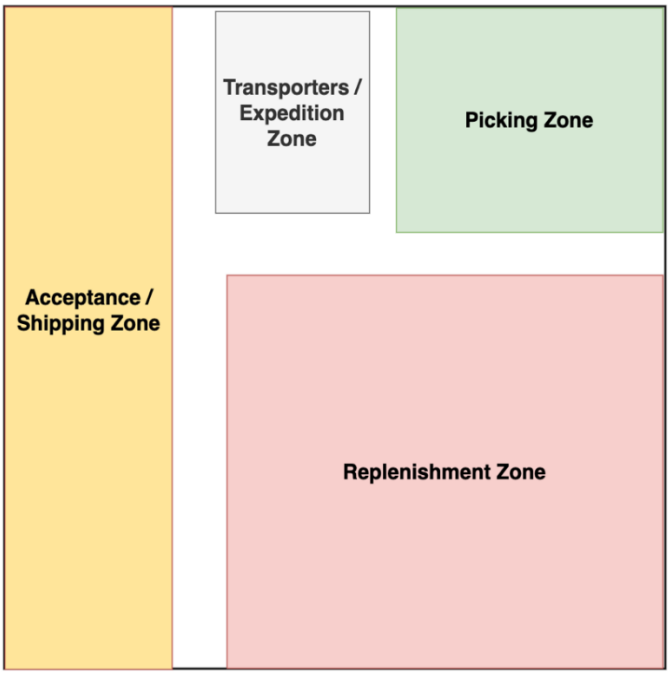

Figure 1: Typical warehouse zones

In the scope of this article, we will concentrate on the acceptance and shipping zones evaluation. To simplify a bit explanation of calculation methodology, let's accept next things: 1) the goods arrive

palletized and packaged in trucks at the warehouse; 2) pallets are homogeneous; 3) goods are accepted after full unloading of cars; 4) the time for acceptance of goods corresponds to the time of car unloading; 5) there is no pronounced trend to increase/ decrease the warehouse stocks; 6) there are no special requirements for storage, handling, commodity management; 7) the storage pallet parameters correspond to the pallet acceptance parameters; 8) orders are shipped after a full check by the forwarding agent of their compliance with the route [10], the time for checking the routes corresponds to the vehicle loading time. 9) the zones with special conditions for commodity flow storage and processing are not required; 10) the works on unloading/acceptance of the goods and the works on order shipment are performed at different times. Consequently, to save resources and warehouse space, it is advisable to use a combined acceptance/shipping zone.

3. Acceptance and shipping zones evaluation

To calculate the required capacity of the acceptance/shipping zone, as well as the required resources, it's required to determine the composition of the first acceptance/shipping post and calculate the required number of posts. Since the loading and unloading fronts are combined, the indicators are calculated separately for the incoming and outcoming commodity flows with the subsequent comparison of the data obtained and the adoption of the highest values.

Let's determine the required number of acceptance and shipping posts. To do this, firstly we need to calculate the number of vehicles arriving per day for unloading, considering the non-uniformity of deliveries. The daily number of vehicles arriving for unloading is determined by the formula:

$$M\ in.V. = \frac{(Vi.o.\ \times\ Rin.)}{(\mathrm{H}p.\ \times\ Sp.\ \times\ Fi.)}, \quad (1)$$

where *Vi.o.* – is an average daily volume of commodity flow, represented in m3,

Rin. – input goods storage unevenness ratio,

Hp. – height (m) of goods on the pallet,

Sp. – area (m2) occupied by the pallet,

Fi. – number of pallets that could be placed in the vehicle that arrive.

Using calculated number of income vehicles (*M in.V.*) from (1) and selected unloading procedure options we can calculate the required number of posts for processing the incoming commodity flow:

$$M\ in.G. = \frac{(Min.V.\ \times\ t\ unload.)}{\mathrm{T}in.}, \quad (2)$$

where *M in.G.* – is the number of parallel a processing incoming commodity flows or gates in the warehouse needed to fulfill acceptance zone requirements,

t unload. – is a time (minutes) for the arrived vehicle unloading, considering technological downtime and auxiliary time,

Tin. – is a scheduled processing time interval of unloading and acceptance of goods.

It's important to mention, we accept that the goods arrive palletized and packaged in trucks at the warehouse, pallets are homogeneous, accepted after full unloading of vehicle. The time for acceptance of goods corresponds to the time of car unloading.

Evaluation of output goods flow is similar with input, income goods. Determine the required number of shipping vehicles using next formula:

$$M\ out.V. = \frac{(Vi.o.\ \times\ Rout.)}{(\mathrm{H}o.\ \times\ So.\ \times\ Fo.)}, \quad (3)$$

where *Rout.* – output goods storage unevenness ratio,

Ho. – height (m) of goods on the packed order's packaging unit,

So. – area (m2) occupied by the packed order's packaging unit,

Fo. – number of packaging units that could be placed in the shipping vehicle.

Using calculated number of outcome vehicles (*M out.V.*) and selected shipping procedure options we can calculate the required number of posts for processing the outcoming commodity flow:

$$M\ out.G. = \frac{(M\ out.V.\ \times\ t\ ship.)}{Tout.}, \quad (4)$$

where *M out.G.* – is the number of parallel processing outcoming commodity flows or gates in warehouse needed to fulfill shipping zone requirements,

t ship. – is a time (minutes) for the vehicle loading, considering technological downtime and auxiliary time,

Tout. – is a scheduled processing time interval of order packaging and shipment of goods.

Usual practice for optimizing warehouse space utilization is to combine input and output gates. For example, if the calculation results in 3 input gates and 10 output gates, then the resulting combination could be: 7 sets of dock equipment (sectional gates, dokshelter, dockleveler) for servicing the low-tonnage vehicles and 3 sets of dock equipment (sectional gates, dokshelter, dockleveler) for servicing both low-tonnage and large-tonnage vehicles. For such a scenario, critical condition added – output and input work shifts shouldn't be overlapped.

Let's determine the required acceptance and shipping zones capacities. Firstly, need to note, the batch of goods is accepted after full vehicle unloading and the acceptance time of the batch of goods corresponds to the time of vehicle unloading. Therefore, to ensure process continuity in the zone, it is advisable to unload the next batch during the acceptance of the batch of goods. To ensure work performance according to this approach, the capacity of one acceptance post should allow us to place a commodity volume equal to double the volume of goods in the vehicle back at a time. This is represented by the next formula:

$$N\ i.p. = 2\ \times\ Fi., \tag{5}$$

Thus, the required zone and capacity of the acceptance post will be calculated using next formula:

$$V\ acc.\ =\ N\ i.p.\ \times \mathrm{H}p.\ \times\ Sp., \tag{6}$$

Finally, the storage space for acceptance zone per one single gate calculated by next formula:

$$S\ acc.\ = \frac{(N\ i.p.\ \times\ Sp.)}{U\ acc.}, \tag{7}$$

where *U acc.* – utilization level of acceptance zone.

By multiplying the obtained values from formulas (5) - (7) by the required number of unloading and acceptance posts, we obtain the required characteristics of the general acceptance zone.

Shipment zone calculation is like the acceptance zone definition. Complete orders in the route are placed in front of the gate. Since the order transfer time to the forwarder corresponds to the order loading time on the back of vehicles, the required capacity and area of the shipping sector through one gate will be as follows:

$$N\ o.p. = 2\ \times\ Fo., \tag{8}$$

$$V\ ship.\ =\ N\ o.p.\ \times \mathrm{H}o.\ \times\ So., \tag{9}$$

$$S\ ship.\ = \frac{(N\ o.p.\ \times\ So.)}{U\ ship.}, \tag{10}$$

where *U ship.* – area usage ratio of shipping zone.

By multiplying the obtained values from formulas (8) - (10) by the required number of shipping posts, we obtain the required characteristics of the general shipment zone.

As usually acceptance and shipping zones are overlapped, it is enough to use the general combined zone space using the maximum obtained value:

$$S\ acc./ship.general\ = max(S\ acc., S\ ship.), \tag{11}$$

Gates are placed at one side of the warehouse with step according to gate size. The diagram of the acceptance/shipping zone is presented in Figure 2. In different literature, this zone could also be named as inbound / outbound zone, income/packaging & sending zone, etc.

The general big square is a building available for design of warehouse, the left side of which is marked for calculated overlapped acceptance / shipping zone. On the left border of the building there are gates with vehicles, from example described early, when we have 3 input gates and 10 output gates, so here we have mixed types of vehicles for different directions of commodity flows.

Now let's assume that as input we have a warehouse building with its height, width, and length. We can arrange gates from any side of a building, also we have a set of 10 different trucks that could be used for delivery of goods to warehouse and for shipping orders. Each vehicle has its own constraints,

like space, loading space width. In the warehouse we have 5 different types of forklifts to use for execution of orders, moving products from packing zone to shipping zone that influence execution time and aisles width. Of course, there is a bigger number of constraints, but we will use only limited, for example: 1) building with possible side options to arrange gates (4 combinations); 2) separate or combined acceptance and shipping zones (2 combinations); 3) acceptance vehicle type (10 combinations); 4) shipping vehicle type (10 combinations); 5) transporter type (5 combinations).

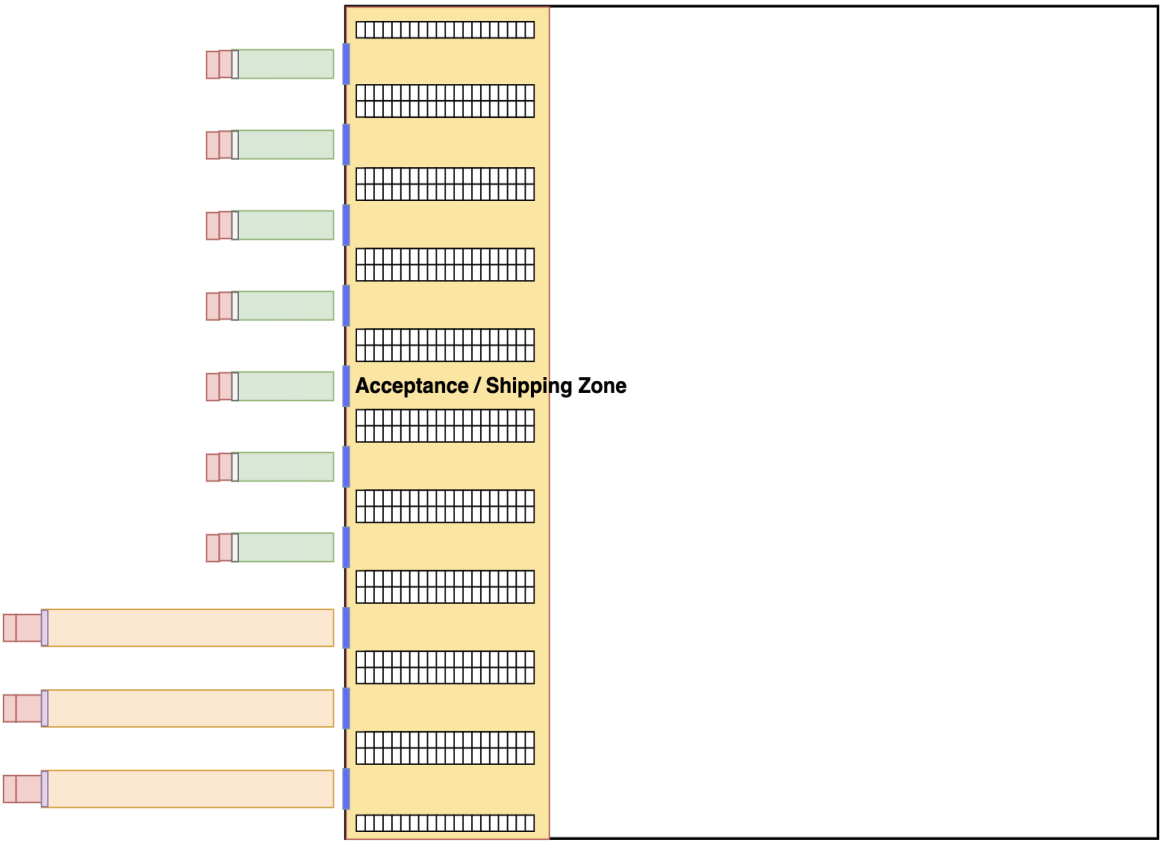

Figure 2: Application of the required areas of the acceptance/shipping zone in the general scheme of the warehouse

To evaluate each combination of parameters that brings us a design solution of acceptance and shipping zones. We can calculate the total expenditure incurred by employees that combine the number of required employees and spent time on order execution – evaluated value is so called "labor cost" [4]. To achieve optimal labor cost value, we need to check all possible combinations of parameters. In the described example, it will be 4 x 2 x 10 x 10 x 5 = 4000 combinations. To check them all manually in real design tasks is a waste of time for engineers because customers wouldn't wait so much time [3], faster result – more chance to success.

4. System architecture and test results

The input data for each warehouse design task, excluding specific conditions and constraints, always contains order data – orders and products. Each order consists of id, shipping due date, target client info, some specific customer properties and a list of order positions, pair product and its quantity. This data could be extrapolated from real data for the last operational year or created virtual with some expectations and forecastings. Using this data, we can calculate input and output good volume flows, unevenness ratio and other criteria required for internal operational calculation.

The architecture of a system at the top level is demonstrated in Figure 3. Input data is uploaded to a sub-system of data and limits processing. As a result, we receive calculated statistics about input data and potential pattern [11], that could be used in the optimal result prediction sub-system for searching

for similar evaluated already warehouse design tasks to find out potential optimal combinations of parameters.

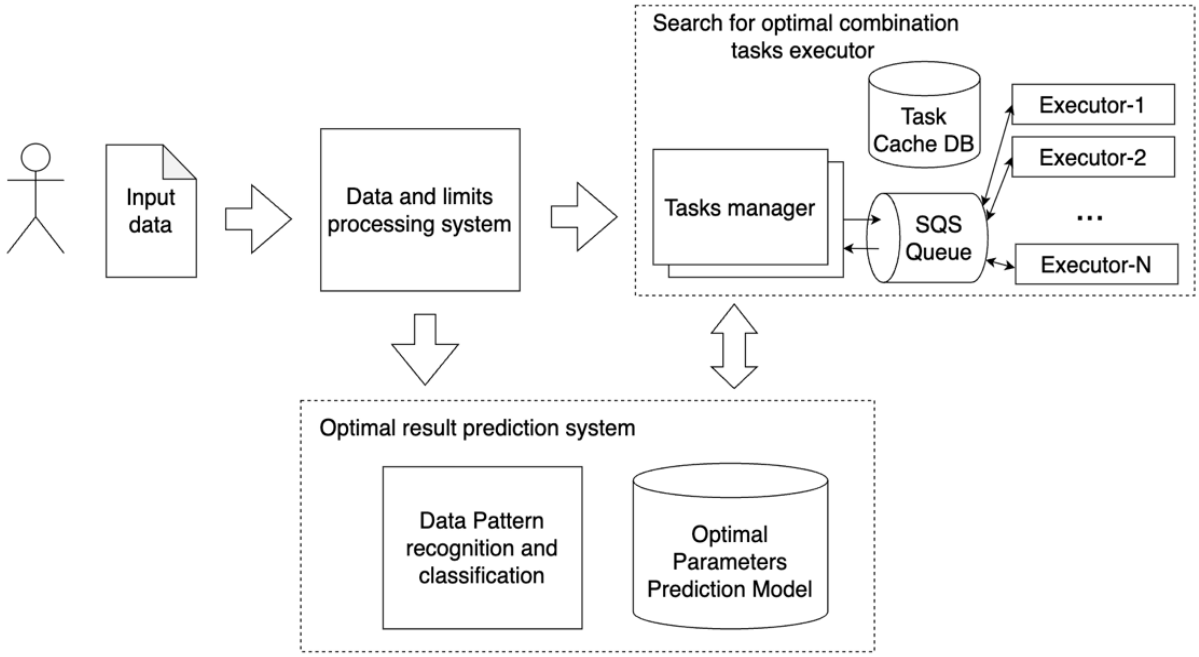

Figure 3: The architecture of developed warehouse design system

Processed input data with goods volume value as an ordinate and time grouped by fiscal quarters as an abscissa for 200000 orders per year totally are shown in Figure 4. Here we can see goods volume fluctuations – unevenness ratio, load peaks and dependency on seasons and important dates. It's very important to consider this during warehouse design tasks to avoid creating battlenacks in operational flows or delays in shipping orders as it is not acceptable from a business point of view.

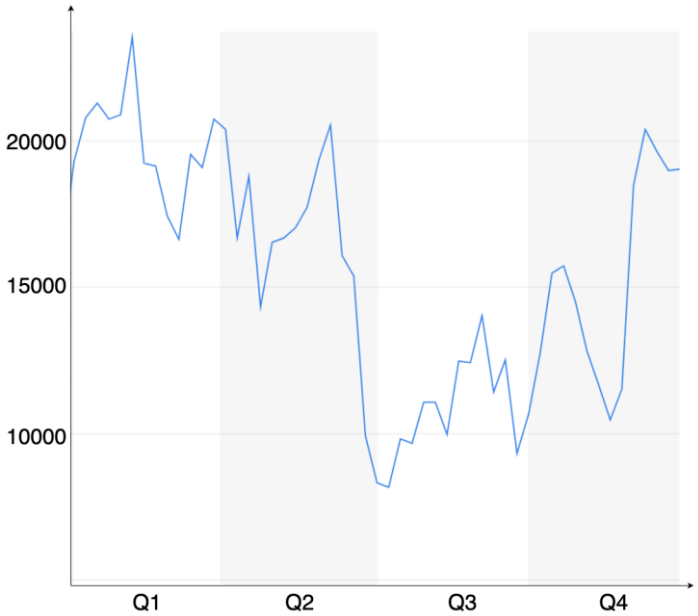

Figure 4: The fluctuations of orders volume (m3) during fiscal year 4 quarters. Example data

The most sensitive sub-system to performance is the search for an optimal combination task executor (see Figure 3), which should use prepared and processed input data and constraints with a list of parameter combinations. The aim is to run an evaluation of each combination – find its summary labor cost value, then define which combination is the most optimal – its labor cost value is minimal.

For that purpose, the task manager caches processed input data and constraints in-memory task cache DB and creates tasks for execution in SQS queue (Amazon Simple Queue Service) [7]. The N executors

triggers execution when events are fired via queue, so the number of light-weighted functions with cached context that evaluate separate combination correlates with the number of task complexity.

We achieve 2 main benefits using serverless computing: saving of resources – we don't use any resources in case we have no active tasks; maximum efficiency of used resources – we run the required number of executors used at max ratio of cpu and RAM load exactly for task execution.

But more interesting is the reduction of execution time for warehouse design tasks, which can be demonstrated by the diagram in Figure 5. Here we have tested different architecture approaches used for executing the same warehouse design tasks with input data of 100000 orders with up to 500000 products and different sets of combinations for each run: 32, 64, 100, 200, 500 and 1000 (you can see them as a group on the abscissa of diagram). The ordinate of the diagram is the time of execution of warehouse design tasks, evaluating all combinations for optimal labor cost search, measured in seconds.

So, architecture approaches from left to right are: 1) simple monolith, here we have a regular 32-core single virtual machine (VM); 2) microservices architecture with horizontal auto-scaling depends on the load that appear during increasing number of combination tasks for evaluation in task queue, using a set of 10 VMs, each one identical to single 32-core VM; 3) serverless computing (running up to 1000 parallel function executors).

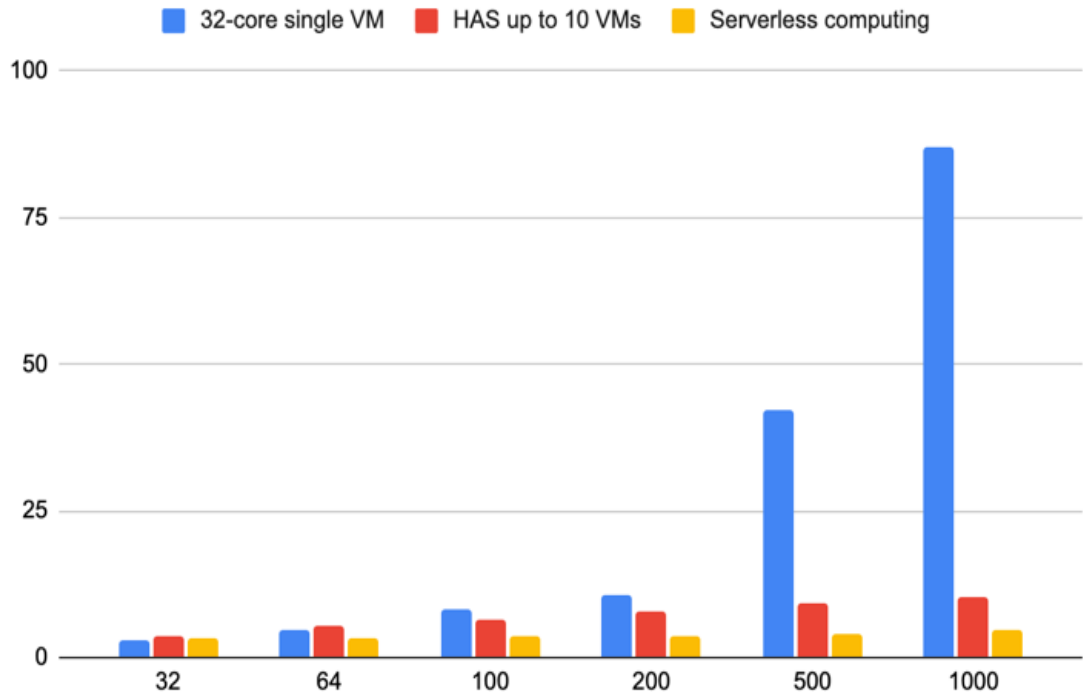

Figure 5: Comparison of warehouse design task execution using different architecture approaches

What we can see from graphics is that, in the small number of combinations (up to 32, till fully load cores of a VM) the leader is for sure a monolith, it's up to 1.5-2 times faster than i.e., microservice architecture because of start-up, initialization context, proxy, and other utility efforts. But starting from 100 combinations, the approach with monolith becomes the looser and at the 1000 combinations we have nearly 1.5 minutes for it, compared to less than 11 seconds using microservices architecture and nearly 5 seconds using serverless computing.

For better comparison of execution time there is a helper graphic in Figure 6 with the percentage difference of time execution between different architecture approaches, marked at the top of figure. The dash line demonstrates comparison of serverless computing architecture to monolith and the dotdash line – serverless computing architecture to microservices with auto-scaling. The abscissa of the graphic is the scope of measured test cases for the described number of combinations, till 1000.

As we can see from graphic, at 1000 combinations load serverless computing is up to 55% faster than microservice architecture and up to 94% faster than single monolith computing. But speed isn't the only the main criteria important for business, another one option is to save money. The serverless architecture is the best choise for that purpose, while there is no computation – there is no need to keep

computing resources running. In Figure 7 there is a demo from Deopware.com how developed system's user interface looks like. The typical user – is a logistic engineer 3PL. On demo picture several simple evaluated design scenarios are shown with the list of all scenarios with costs.

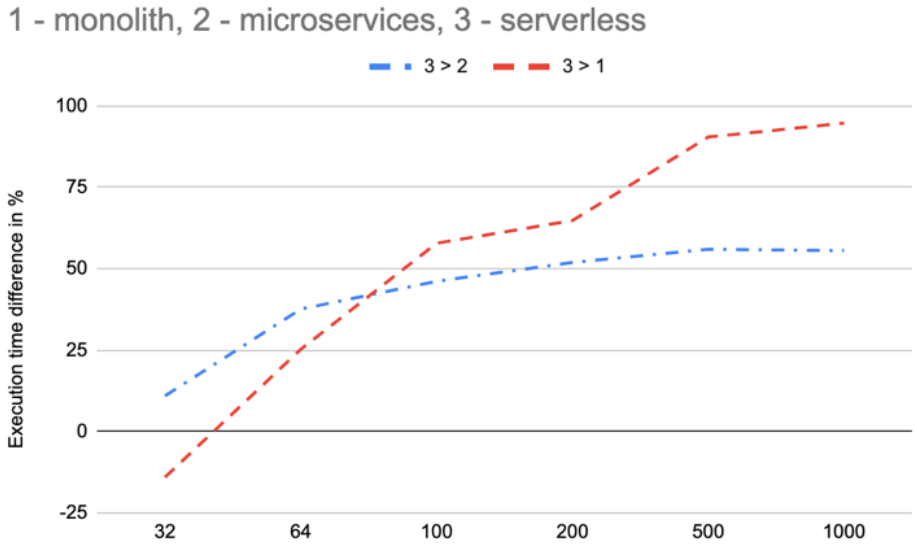

Figure 6: Percentage difference in time execution of serverless computring compared to monolith and microservices approaches

#	Stage	Scenario Name	Sample Rack Name	Sample Storage Unit Name	Bill of Materials Name	Storage Unit Volume [m³]	Total Storage Unit Volume [m³]	Storage Unit Area Size [m²]	Number of Storage Units	Number of Racks	Total Price	Number of Products	Updated Date	
1	3	2019 inventory	PR 350_2pall pl	PR 350_2pall pl-1-1	PR 350_2pall pl	1.35	0.0	1.08	13	4	800.0	5	2019-10-02 17:17:54.0	✕
2	3	Advent period	PR 350_2pall pl	PR 350_2pall pl-1-1	PR 350_2pall pl	1.35	0.0	1.08	2	1	200.0	5	2019-10-01 19:17:38.0	✕
3	2	Advent period	Shelving_T1	Shelving_T1-2-1	Shelving T1	0.1148	0.0	0.27	1	1	65.0	1	2019-10-01 19:17:38.0	✕
4	2	2019 inventory	Medium-Duty Shelving5	Medium-Duty Shelving-1-2	BOM Medium-Duty Shelving	0.4806	0.87837607813	0.81	15	5	450.0	5	2019-10-02 17:17:54.0	✕
5	2	2019 inventory	Shelving_T1	Shelving_T1-2-1	Shelving T1	0.1148	0.0	0.27	0	0	0.0	0	2019-10-02 17:17:54.0	✕
6	2	Advent period	Medium-Duty Shelving	Medium-Duty Shelving-1-2	BOM Medium-Duty Shelving	0.4806	0.0	0.81	4	2	180.0	4	2019-10-01 19:17:38.0	✕
7	3	Advent period	PR 350_3pall pl	PR 350_3pall pl-2-1	PR 350_3pall pl	1.35	0.0	1.08	0	0	0.0	0	2019-10-01 19:17:38.0	✕
8	3	2019 inventory	PR 350_3pall pl	PR 350_3pall pl-2-1	PR 350_3pall pl	1.35	0.0	1.08	0	0	0.0	0	2019-10-02 17:17:54.0	✕

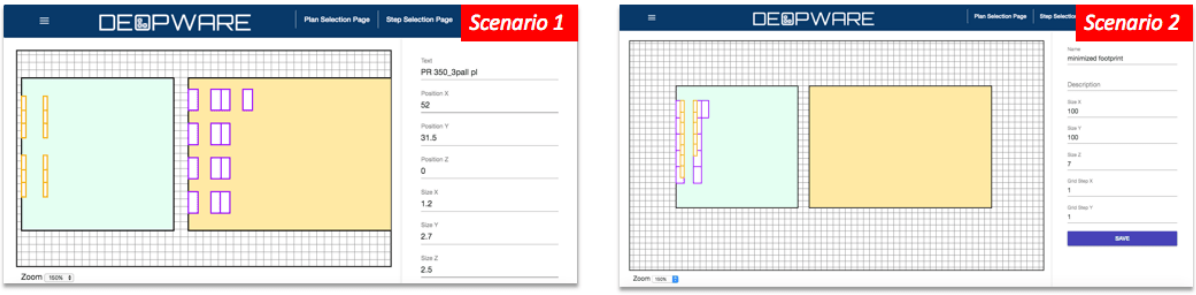

Figure 7: Demo of used developed system for warehouse design (*source http://deopware.com/)

Described developed system used by Deopware.com with multi-tenant environments that are deployed on popular cloud providers (Google, Amazon) because of serverless computing brings up to 65% of resources savings per month. The network part that includes dedicated IP addresses is billed on a permanent basis constantly, the same as storage of data. But CPU, RAM, and everything else using serverless computing is fully dependent on the load that customers create. For demonstration, there is a snapshot of resources savings in Figure 8. In percentages on the ordinate is value of resources saved using a serverless approach comparing to equal regular set of VMs in GCS (Google cloud provider services). The statistic was accumulated for one of the environments – tenant-1 in production during 30 days, presented on the abscissa.

In addition, using serverless computing there is possibility to determine calculation time, faster client need to achieve result, more executors would be used, with more costs per hour. To demonstrate

performance of system used by Deopware.com, in Table 1 is shown a set of tasks executed using different scaling ratios.

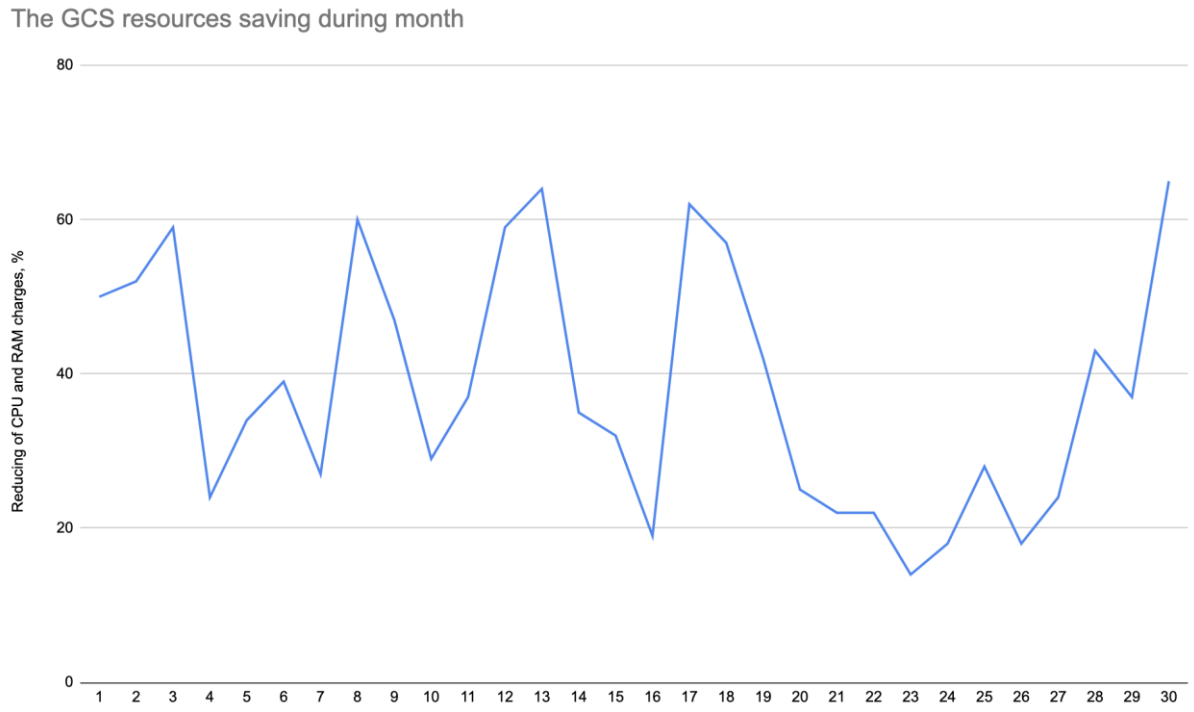

Figure 8: The resource saving in GCS 1-st tenant for 30 days.

Table 1
Demonstration of waregouse design task calculations

Task Name / Type	100-executors	500-executors	1000-executors
Task1 (3128 combinations)	2146 s	486 s	299 s
Task2 (5980 combinations)	4144 s	949 s	572 s
Task3 (6130 combinations)	4506 s	993 s	587 s
Task4 (7005 combinations)	4805 s	1123 s	644 s
Task5 (7761 combinations)	5704 s	1220 s	764 s
Task6 (8608 combinations)	5724 s	1394 s	808 s
Task7 (8890 combinations)	6659 s	1469 s	875 s
Task8 (10219 combinations)	7225 s	1589 s	940 s
Task9 (21010 combinations)	15148 s	3404 s	1933 s

The main characteristic of each task is prepared number of parameters and constraints combinations for evaluation and searching the optimal one by labor cost. In the list of commercial tasks, per each of them are demonstrated execution times required for full combination evaluation solved on different scaling of serverless computing: 100, 500 and 1000 executors. The range of combinations in tasks

approximately from 3000 till 21000. Using executors pool scaling from 100 to 1000 the calculation time of tasks could be reduced in 7 times and more.

5. Summary

The developed system for automated warehouse design using serverless computing and prediction model enables reducing time for search of parameter's combination that will help to achieve optimal labor cost value in acceptable time comparing to manual evaluation of several selected potentially good combinations, that highly boosts the performance of 3PL provider companies.

The selected computing architecture has the main impact on performance. Tested three different approaches: a monolith single virtual machine, microservices with horizontal auto-scaling and the serverless architecture. The results demonstrate that serverless computing is slower on low complexity tasks with parallel processing threads nearly 1.5 times higher than the number of single VM's cores. But starting from medium and high complexity compared to other two variants, the advantage become obvious. Moreover, the serverless computing approach is less expensive and more resources saved compared to microservices even with autoscaling, because of fast startup and less context size.

6. References

[1] S. S. Heragu, Facilities Design. CRC Press, 2018, pp. 261–314.

[2] A. Ekeskar, Exploring Third-Party Logistics and Partnering in Construction: A Supply. Linköping University Electronic Press, 2016.

[3] K. Grzybowska, A. Awasthi, R. Sawhney, Sustainable Logistics and Production in Industry 4.0: New Opportunities and Challenges. Springer Nature, 2019, pp. 31–57.

[4] M. P. Stephens, Manufacturing Facilities Design & Material Handling, 6-th ed.. Purdue University Press, 2019, pp. 287–301.

[5] C. M. Bishop, Pattern Recognition and Machine Learning. Springer New York, 2016, pp. 179–196.

[6] L. Heng, B. McColl, Mathematics for Future Computing and Communications. Cambridge University Press, 2021, pp. 309–329.

[7] D. Poccia, AWS Lambda in Action: Event-driven serverless applications. Simon and Schuster, 2016, pp. 287–334.

[8] P. Raj, S. Vanga, A. Chaudhary, Cloud-native Computing: How to Design, Develop, and Secure Microservices and Event-Driven Applications. John Wiley & Sons, 2022. – pp. 129–163.

[9] R. Walter, N. Boysen, A. Scholl, "The discrete forward-reserve problem – allocating space, selecting products, and area sizing in forward order picking", European Journal of Operational Research, vol. 229(3), pp. 585–594, 2013.

[10] O. Muliarevych, "Solving dynamic asymmetrical Travelling Salesman Problem in conditions of partly unknown data", Lviv-Slavsk, Lviv Polytechnik Publ., TCSET'2016 vol.1, pp. 446–448, February 2016.

[11] G. Bonaccorso, Machine Learning Algorithms, 1st ed.. Packt Publishing, 2017, pp. 476–507.