Vespa.ai

We Make AI Work

The End of Overprovisioning: How Vespa Automates Growth Without Wasting Resources

Autoscaling with Vespa Cloud

The End of Overprovisioning:

How Vespa Automates Growth Without Wasting Resources

Introduction

Businesses frequently face fluctuating workloads, whether driven by seasonal peaks, sudden traffic surges, or consistent growth. Media sites, for example, must handle viewer spikes during live events or viral content without compromising user experience. Similarly, retail platforms need to seamlessly manage seasonal demand increases or user-generated data, as seen on platforms like Vinted.

In these scenarios, maintaining optimal performance while controlling costs is essential. This eBook explores how Vespa's advanced autoscaling capabilities help organizations efficiently manage variable workloads by automatically adjusting resources to meet performance, cost, and scalability requirements.

How Autoscaling Works in Vespa

Vespa's unique **autoscaling** feature works by monitoring system metrics such as CPU and memory usage. If resource usage goes above the calculated ideal load for the cluster in question, the system adds capacity to keep resources in the optimal range. Likewise, when demand drops, the system scales down to avoid unnecessary costs. Vespa stores and processes data in servers called **nodes**, which belong to **clusters**, which ensure reliability and scalability in both data size and parallel processing capacity.

Container nodes process incoming queries, manage workload distribution, and support tasks like creating embeddings and interfacing with large language models (LLMs). These lightweight nodes can scale quickly to meet demand.

Content nodes store and process data, efficiently handling read, query, and write operations.
Now that we've covered the fundamentals, let's explore different scaling strategies and how they impact performance and cost.

Scaling Strategies: More Nodes vs. Bigger Nodes

When deploying search, recommendation, or AI-powered applications, one of the key challenges is scaling resources efficiently. Should you add more nodes to handle increased traffic, or should you increase the size of nodes to accommodate internal changes to your application? This decision directly impacts performance, cost, and flexibility.

Scaling by the number of nodes

Adding more nodes offers more flexibility, allowing easy allocation or removal of resources. This method is ideal for handling short-term traffic spikes or fluctuating workloads, and it keeps costs predictable and linear, as adding nodes directly correlates with increased load. However, what truly reduces cost is automatic downscaling. Vespa not only adds capacity when demand increases but also removal of excess nodes when demand drops, ensuring you don't pay for unneeded resources.

When setting up autoscaling, you can set a range of nodes (e.g., <nodes count=[2-4]>). Vespa will automatically determine the optimal number of nodes within this range and distribute data seamlessly based on the current CPU load, used memory size, and used disk size. For example, during a surge in traffic, Vespa quickly adds capacity to handle increased activity and keep response times low. When demand decreases, Vespa automatically reduces capacity, cutting costs while staying ready for future spikes.

Vespa keeps node utilization within safe ranges to ensure smooth performance. Unlike traditional Lucene-based clusters that require fixed shards, Vespa can seamlessly redistribute data and remove unnecessary nodes, eliminating the need for manual intervention.

With both upscaling and downscaling, businesses can reduce expenses, avoid unnecessary infrastructure costs, and maintain fast performance—all while allowing Vespa to intelligently adjust resources as needed.

Scaling by changing node type

As an alternative to scaling the number of nodes you can also choose to change the type of nodes used. For a given workload and (for stateful nodes) data set, there is a particular ratio between cpu, memory and disk which has the best fit in the sense of utilizing each resource to the maximum extent possible.

This determines the node family within the cloud provider. In addition, each node family comes in many sizes, where each node resource is scaled by the same amount, keeping the ratio constant.

Lastly, each family often comes in multiple generations, where newer generations provide better performance per resource unit. This dimension can be ignored with Vespa Cloud since the system will automatically pick the newest generation available, and migrate clusters to newer generations as they become available.

The optimal node family must be determined by observing resource usage, and the optimal node size to use is a tradeoff between needing one redundant node (indicating smaller nodes being optimal), and index lookup efficiency being sub-linear (indicating larger nodes).

Using the right node type will substantially save the cost of clusters, but can be challenging as resource needs change over time with changes in data, and queries and writes. The drawback of scaling by changing node type of stateful clusters is that all data need to migrate to the new nodes, which while safe to do while serving can take a long time during which both node types are active.

In any case, autoscaling can automatically determine these tradeoffs, including the time to migrate, and pick the right node type to use at a given point in time by specifying ranges for node resources in addition to node count.

Scaling by changing the number of groups

In addition to changing the number of nodes, and the size of resources of each node, content clusters support a third scaling dimension: The number of groups.

Each group in a content cluster contains one or more complete copies of all the data indexed in the cluster. By default, there is a single group, meaning content is distributed over all the nodes. This means that all queries must be distributed to all the nodes in the cluster. This is fine at lower query rates, but is economical when growing to large query volumes. By configuring multiple groups, each query only need to be routed to a single group in the content cluster, allowing you to scale efficiently to any amount of query load.

The number of groups in a cluster can be set as a range in Vespa just like the number of nodes and node resources, which allows Vespa to scale the number of groups up and down to match the current amount of traffic. Scaling groups may be more efficient than scaling nodes or node resources, as it does not involve redistributing data between existing nodes.

Business Continuity Planning (BCP)

A key to ensuring that applications remain available and operational even when unexpected failures occur is deploying across multiple regions. This approach safeguards against potential failures in any single zone, whether due to regional issues—such as power outages or network disruptions—or planned sequencing, like staged feature rollouts across different zones. By distributing workloads across multiple locations, businesses can enhance reliability, minimize downtime, and maintain seamless user experiences.

Vespa allows you to specify that your application should be deployed in any collection of regions (across AWS, GCP and Azure), and to specify endpoints that are global across regions. With this configured, your application is not dependent on any single region being available to stay live. BCP increases availability but also cost. Autoscaling comes into play here by allowing you to make a more flexible tradeoff between availability and cost:

You can configure backup regions that are unable to take the additional load from a region being down and rely on autoscaling to scale up capacity only when there is a regional outage.

Autoscaling makes this simple by requiring you to only specify the deadline for having full capacity online elsewhere after a regional outage, and let the Vespa Cloud autoscaling to optimize resource allocation to achieve that as cost effectively as possible.

Tailoring autoscaling to your workload

Vespa makes it easy to manage multiple clusters, giving you the flexibility to handle different workloads efficiently. For example, you can set up a dedicated cluster for batch-feeding tasks that scale independently of your query traffic. This helps keep workloads isolated, and ensures resources are used effectively.

Imagine a retail platform periodically updating its product catalog. The catalog updates are uploaded in large batches, requiring high system capacity. With Vespa, a dedicated autoscaled cluster could efficiently handle the batch feed without impacting the user-facing query cluster. The batch cluster scales up when the update begins and scales down once it is complete, ensuring the operation is fast and cost-effective.

Conclusion

Vespa's stateless and stateful autoscaling ensures that users handle high-throughput, low-latency workloads with minimal manual intervention while keeping systems resilient and responsive, so that:

- Performance remains stable even as demand fluctuates.
- Data consistency is preserved during scaling events.
- Infrastructure costs are optimized by dynamically managing resources.

Vespa ensures that businesses can optimize costs, ensure high performance, and seamlessly scale with demand.

Vespa.ai is a platform for building and running real-time AI-driven applications for search, recommendation, personalization, and RAG. It enables enterprise-wide AI deployment by efficiently managing data, inference, and logic, handling large data volumes and over 100K queries per second. Vespa supports precise hybrid search across vectors, text, and structured metadata. Available as both a managed service and open source, it's trusted by organizations like Spotify, Vinted, Wix, and Yahoo. The platform offers robust APIs, SDKs for integration, comprehensive monitoring metrics, and customizable features for optimized performance.

Interested to learn more? We have many different resources and information available through our social platforms

GitHub Twitter LinkedIn YouTube