Vespa.ai

Balancing Performance and Cost: A Guide to Optimizing Node Size in Vespa

We Make AI Work

Vespa.ai

We are 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

Contents

Document
Balancing Performance and Cost: A Guide to Optimizing Node Size in Vespa

2025

↗ All contents contain linked titles.

Introduction

The Challenge of Post-Deployment Optimization

Achieving the perfect balance between cost, performance, and quality is a complex, ongoing process that requires more than just identifying problem areas. It demands continuous improvement, informed decision-making, and the ability to adapt quickly to new demands. Vespa simplifies this journey, providing the tools and insights needed to optimize systems, ensure stable deployments, and keep operations running smoothly—all while freeing teams to focus on innovation.

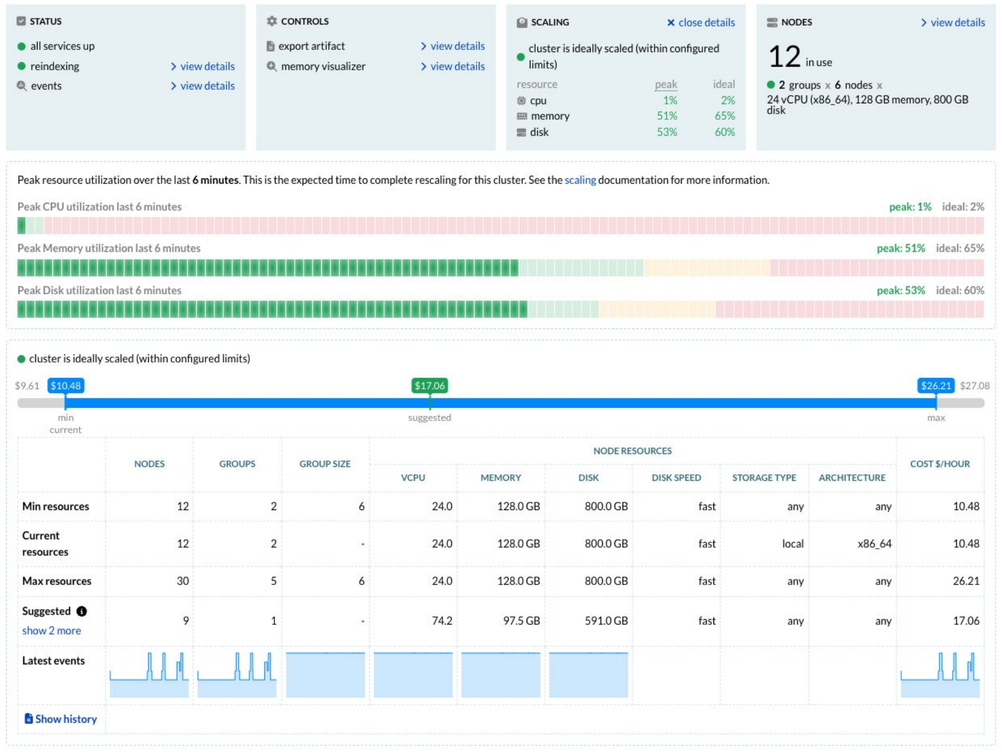
This ebook explores how Vespa helps organizations navigate the complexities of system optimization, scale resources effectively, and achieve their business goals without compromising the end-user experience.

Deploying a new system rarely results in the perfect configuration right away. Achieving the ideal balance between cost, performance, and quality requires addressing trade-offs. For example, reducing costs might degrade performance and quality, while metrics like latency and throughput can decline during traffic surges or configuration changes.

Different organizations approach these trade-offs based on their priorities. For example, companies like Netflix prioritize performance and might overprovision resources to deliver predictable low latency. Meanwhile, cost-conscious startups may tolerate occasional latency spikes to save resources and reduce costs.

This balancing act begins with identifying areas for improvement, making iterative adjustments, and monitoring performance continuously. Optimization isn't a one-time task—it's an ongoing process of refining and adapting to meet evolving needs.

Automation and Suggestions: Bridging the Gap



Vespa simplifies the optimization process, empowering organizations to evaluate the impact of changes quickly, iterate confidently, and deploy updates without risking production stability. By monitoring system performance and providing actionable suggestions, Vespa ensures that changes are implemented intelligently and without disrupting production.

Vespa's platform provides two critical perspectives to cater to different user needs: High-Level Overview: Aggregates key metrics such as latency, request rates, and resource utilization. This perspective offers actionable insights for business stakeholders and developers, making aligning technical decisions with strategic goals easier.

Detailed Analytics: For experts who require in-depth analysis, Vespa provides detailed dashboards with granular data. These analytics enable precise adjustments to system configurations, though interpreting them effectively often requires proper guidance. The Vespa Console offers guided optimization for critical resources like CPU, memory, and disk space, ensuring efficient resource allocation.

CPU: Plays a key role in both container and content nodes. For container nodes, it handles tasks like query parsing and result aggregation. On content nodes, more CPU power reduces latency by allowing multithreaded execution and increasing capacity per query. It also boosts overall throughput by adding more virtual CPU cores to manage higher loads.

Memory: Primarily essential for content nodes, as it is closely tied to data size and index settings. While less critical for container nodes, large models can be deployd to both and take space in memory.

Disk Space: Required for persistence to keep the raw data and the index structures.

With Vespa, you can make these adjustments intelligently and iterate quickly to maintain optimal system performance.

Making System Improvements

Dynamic Scaling: Auto-Scaling vs Fixed Scaling

Vespa uses a powerful combination of automated tools and intelligent suggestions to help you optimize system performance:

Automated Recommendations: Based on traffic patterns and historical data, Vespa can suggest specific changes, such as resizing nodes, adding capacity, or redistributing workloads, to improve efficiency and scalability.

Incremental Adjustments: Gradual, smaller changes are safer and allow systems to adapt without production disruptions. For instance, reducing the number of nodes incrementally ensures predictable resource utilization, helping you identify the optimal balance for efficiency.

Whether it's handling data growth, increasing user load, scaling container and content nodes, optimizing for cost, or deploying updates with minimal risk, Vespa provides automation and insights to support these critical tasks.

In Vespa, scaling can be handled dynamically or manually, with both approaches offering their own advantages.

Manual Scaling: A user explicitly decides the amount of machine resources, for example, choosing a different amount of CPU or changing the number of nodes in a cluster. All such changes are handled safely and automatically, and data is redistributed automatically across the nodes without impacting queries or writes.

Auto-Scaling: Instead of fixing a specific amount of machine resources, users can define ranges, e.g., 5 to 10 nodes with between 32 and 128 GB memory. The system will automatically find the cheapest configuration that handles the current load on the system, and change it as needed, ensuring cost efficiency without compromising performance. For content clusters, the time to converge on a new configuration may be significant, which is also considered when determining if a change is worthwhile. range.

This flexibility is a significant advantage over traditional elastic clusters with fixed shards, where shard numbers cannot dynamically change, and redistribution is manual and time-consuming. Auto-scaling also ensures that capacity is always available when needed, preventing the system from reaching a "full" state.

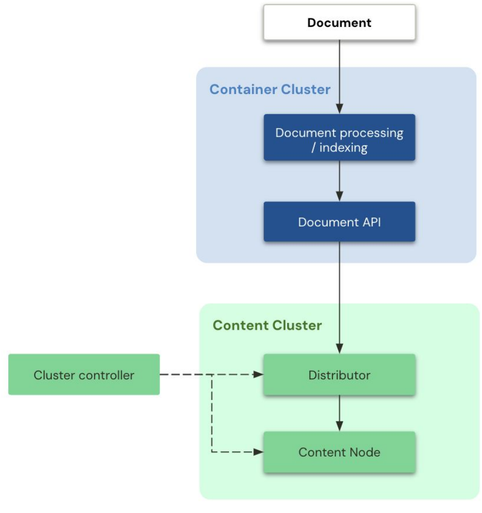
To improve result quality, you can use more advanced ranking models, though they require more resources, creating a tradeoff between cost and quality. Vespa makes it easy to handle these adjustments by scaling smoothly for even the most complex use cases. Whether the data volume changes, user queries fluctuate, or both, Vespa ensures that resources adapt dynamically to stay within the optimal operating

Differences in Scaling for Container and Content Nodes

By distinguishing between the roles of container and content nodes, Vespa ensures that companies can scale efficiently for long-term growth or daily fluctuations.

Content nodes store, index, search, and make inferences in data. Distributing large datasets can take hours, so scaling content clusters takes a longer planning horizon.

In contrast, **container nodes** are stateless and can scale in a few minutes, making them able to track daily traffic fluctuations closely.



Growing Data vs. User Load

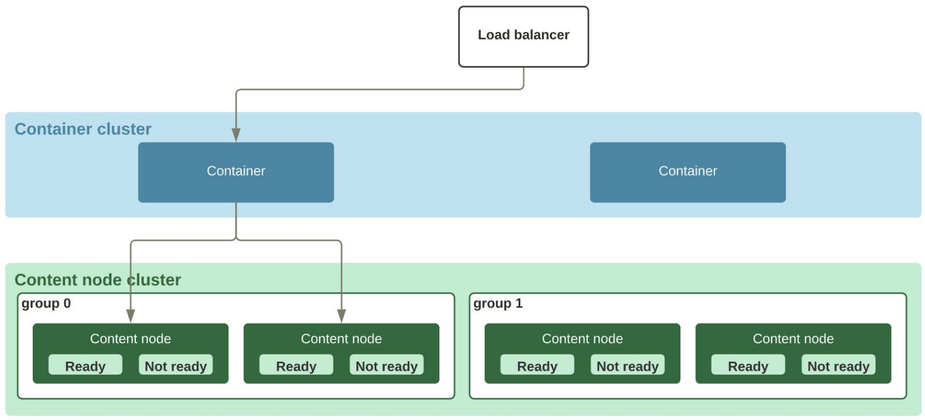
One of the key challenges in scaling is distinguishing between growing data volume and increasing user load:

Data Growth: More products, larger datasets, or increasing memory and disk requirements.

User Load: Higher traffic, more concurrent users, heavier queries, or increased query rates.

Vespa helps analyze performance across nodes, providing actionable insights and suggestions for optimizing configurations. The platform supports adjusting node types, counts, and group layouts to meet evolving needs.

Optimizing Topology



Adding more nodes typically reduces latency by lowering the number of documents per node. Reducing the number of documents per node lowers latency by minimizing the portion of query cost that depends on document count. However, beyond a certain point, the fixed cost of processing queries on each node becomes a significant factor

Vespa's topology optimization balances performance and resource utilization in those cases as well, by organizing content nodes into groups. Each group has a separate copy of all the data, so each query can be routed to just one group, making it possible to scale efficiently to very high query loads.

Vespa's console provides recommendations for achieving the optimal topology based on these principles, and you can optimize cluster topology to handle increases in query volume without impacting production.

Vespa’s Safe Deployment

Pre-deployment Validation

Continuous Testing

Deploying updates to a live system includes risks. Unexpected issues, performance bottlenecks, or resource mismanagement can disrupt services and negatively impact user experience. Here’s how Vespa ensures your updates are reliable and low-risk.

Vespa validates every set of changes before deployment to a running application, ensuring they meet requirements before proceeding to production. In addition to ensuring the integration of each application revision on its own, it is also validated against the currently running system. This ensures that destructive changes are not inadvertently reaching production.

Changes that may be destructive can still be deployed, but they would require an explicit validation override to proceed.

Any set of application changes to be deployed in production is first automatically passed through two test stages:

System testing sets up a downscaled version of the application and runs any system tests defined as part of the application to determine if it should proceed.

Staging testing sets up a downscaled version of the application currently running in production and performs the upgrade to the new application revision to ensure the change itself is also safe.

Together these test phases provide the safety needed to do automatic continuous deployment of application changes. This puts in place the fast feedback loop that is the necessary foundation for excellence and innovation.

Architected for High Availability Under Change

Vespa is designed from the ground up to implement changes seamlessly without disrupting queries or writes. This includes its ability to run multiple versions of custom application components concurrently, enabling atomic switches without restarts. Additionally, its data distribution subsystem maintains correctness during data migrations, along with various other system mechanisms that ensure uninterrupted operation.

As a result, any change that passes pre-deployment validation against your running system can be safely deployed without disrupting queries or write traffic.

Phased Deployment

Even with these safeguards, deploying updates to your entire user base at once can be risky. Vespa lets you define phased rollouts across multiple zones or application instances, allowing you to introduce changes gradually. For example, you might begin by deploying updates to just 5% of users, then expand to 25%, and finally reach 100%. Each phase can include delays and testing, ensuring issues are caught before progressing to the next. This incremental approach helps identify subtle issues early, reducing the risk of widespread impact on your user base.

Incremental Adjustments

Large, sweeping changes can lead to unpredictable results. With automatic, safe continuous deployment, Vespa encourages the practice of making small, incremental changes, which reduces risk and makes issue diagnosis faster and easier.

Making frequent, small changes minimizes risk and fosters a culture of continuous deployment within your organization. Deployments become routine, building confidence in the process and accelerating innovation. As updates become a regular part of operations, time to market decreases, enabling faster delivery of new features and improvements.

Real-time Monitoring

Vespa offers detailed monitoring tools and dashboards to track your system's real-time behavior, including the impact of deployed changes. This allows you to quickly identify shifts in resource utilization, latency, and overall performance.

By combining these capabilities, Vespa minimizes risk, enhances system stability, and ensures a seamless user experience. With powerful monitoring and control, you can confidently deploy updates, knowing you have the tools to manage changes effectively and keep operations running smoothly.

Conclusion: Enabling Innovation while Reducing Risk

Whether adding new features, optimizing for peak traffic, or scaling resources, Vespa ensures that teams can confidently make data-driven decisions. For businesses, this translates into continuous customer experience improvements, a stronger competitive edge by enabling innovation, and the ability to scale cost-effectively without compromising service quality.

Vespa removes the guesswork for engineering teams, enabling them to quickly pinpoint areas that need enhancement, implement the necessary changes, and deploy updates safely and efficiently—all while tracking measurable performance gains. This streamlined process allows for faster and safer iterations, enabling engineers to focus on driving feature innovation with minimized risk.

Managing distributed systems can be a complex and time-consuming task, but it doesn't have to be. By simplifying operations, providing actionable and easy-to-understand insights, and automating repetitive processes, Vespa ensures that your systems are efficient and resilient, enabling you to stay competitive in an ever-evolving landscape.

About Vespa.ai

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

Vespa.ai