Analyzing Priority-Based Temporal Workflow Dispatch under External Rate Limits

Stanislav Kosorin
Matrikelnummer: 457294

Workflow engines such as Temporal are widely used to orchestrate long-running distributed applications through durable execution, task queues, and automatic retries. In many production systems, workflows primarily coordinate calls to external APIs rather than performing substantial computation locally. External providers enforce fixed concurrency limits and rate budgets that are known to operators but enforced outside the workflow engine. Temporal schedules tasks based on queue state, worker availability, and task metadata, without explicit knowledge of downstream provider capacity. Provider overload therefore becomes visible only after a request is rejected, for example through throttling responses.

Temporal provides built-in mechanisms for priority ordering and weighted fairness. Priority keys determine relative urgency, and fairness keys regulate how dispatch capacity is shared across tenants or workload classes when tasks are backlogged. These mechanisms influence ordering and relative shares but do not regulate overall dispatch volume with respect to external provider limits. When throttling occurs, Temporal treats the event as task failure and applies the configured retry policy. Retried tasks re-enter the system with the same priority and fairness metadata and are indistinguishable from first attempts. Under provider saturation, retries consume dispatch capacity without increasing successful throughput, which can lead to elevated retry rates, increased waiting times for lower-priority work, and persistent backlog growth.

These effects are amplified in multi-tenant deployments where multiple tenants share the same external provider. Provider-unaware dispatch couples otherwise independent tenants through the external bottleneck, so retries and backlog growth from one tenant can directly delay work from others. This coupling arises even when tenants use different priority classes and when average arrival rates remain below documented provider limits, since overload is driven by short-term bursts and retry amplification rather than steady-state demand.

The goal of this thesis is to compare provider-agnostic dispatch with dispatch that explicitly accounts for provider limits and observed throttling under fixed priorities and retry semantics. It introduces a provider-aware admission control layer that operates on top of Temporal without modifying its internals. Workflow executions are created normally, but provider-bound activities are scheduled only after explicit admission. Workflows

park while waiting for admission and proceed only when capacity is available. Priority and fairness determine ordering among admitted work, while admission control determines how much work is allowed to proceed. The evaluation measures tail latency per priority and per tenant, retry rates, backlog growth and drain behavior, and worst-case waiting times using replayed production workflows and controlled synthetic load. The expectation is that provider-agnostic dispatch leads to elevated retry rates, cross-tenant interference, and persistent backlog growth under provider saturation, while rate-limit-aware admission reduces retries, limits backlog growth, and lowers tail latency without violating Temporal's priority and fairness semantics.