Cloud Resource Management and Scheduling

Manajemen dan penjadwalan sumber daya

- Fungsi kritis dari setiap sistem buatan. Akan mempengaruhi tiga kriteria dasar untuk evaluasi sistem:
  - Fungsionalitas.
  - Kinerja.
  - Biaya.
- Penjadwalan dalam sistem komputasi → memutuskan bagaimana mengalokasikan sumber daya sistem, seperti siklus CPU, memori, sekunder ruang penyimpanan, I/O dan bandwidth jaringan, antara pengguna dan tugas.
- Kebijakan dan mekanisme untuk alokasi sumber daya terdiri dari:
  - Kebijakan prinsip-prinsip yang memandu keputusan.
  - Mekanisme sarana untuk mengimplementasikan kebijakan.

Tujuan Manajemen sumber daya cloud

- Manajemen sumber daya cloud.
  - Memerlukan kebijakan dan keputusan yang kompleks untuk optimasi.
  - Tantangannya akan memiliki kompleksitas sistem tidak memungkinkan untuk memiliki informasi secara global yang akurat.
  - Dipengaruhi oleh interaksi tak terduga dengan lingkungan, misalnya, kegagalan sistem, serangan.
  - Penyedia layanan cloud dihadapkan pada beban berfluktuasi besar yang menantang klaim elastisitas awan.
- Strategi pengelolaan sumber daya untuk IaaS, PaaS, dan SaaS akan sangat berbeda.

Kebijakan manajemen sumber daya cloud (Cloud resource management (CRM))

1. Admission control → mencegah sistem menerima beban kerja melanggar kebijakan sistem tingkat tinggi.
2. Capacity allocation → mengalokasikan sumber daya untuk aktivasi individu dari sebuah layanan.
3. Load balancing → mendistribusikan beban kerja secara merata di antara server.
4. Optimalisasi energi → meminimalkan konsumsi energi.
5. Quality of service (QoS) → menjamin kemampuan untuk memenuhi waktu atau kondisi lain yang ditentukan oleh Perjanjian Tingkat Layanan.

Mekanisme implementasi sumber daya kebijakan manajemen

- Control theory → menggunakan umpan balik untuk menjamin stabilitas system dan memprediksi perilaku sementara.
- Machine learning → tidak memerlukan model kinerja sistem.
- Utility-based → memerlukan model kinerja dan mekanisme untuk mengkorelasikan kinerja tingkat pengguna dengan biaya.
- Market-oriented/economic → tidak memerlukan model system.

pertukaran

- Untuk mengurangi biaya dan menghemat energi, kita mungkin perlu memusatkan beban pada server yang lebih sedikit daripada menyeimbangkan beban di antara mereka.
- Kita mungkin juga perlu beroperasi pada clock rate yang lebih rendah; kinerja menurun pada tingkat yang lebih rendah daripada energi

CPU speed (GHz)	Normalized energy (%)	Normalized performance (%)
0.6	0.44	0.61
0.8	0.48	0.70
1.0	0.52	0.79
1.2	0.58	0.81
1.4	0.62	0.88
1.6	0.70	0.90
1.8	0.82	0.95
2.0	0.90	0.99
2.2	1.00	1.00

Aplikasi dan konsep kontrol ke sumber daya cloud manajemen (CRM)

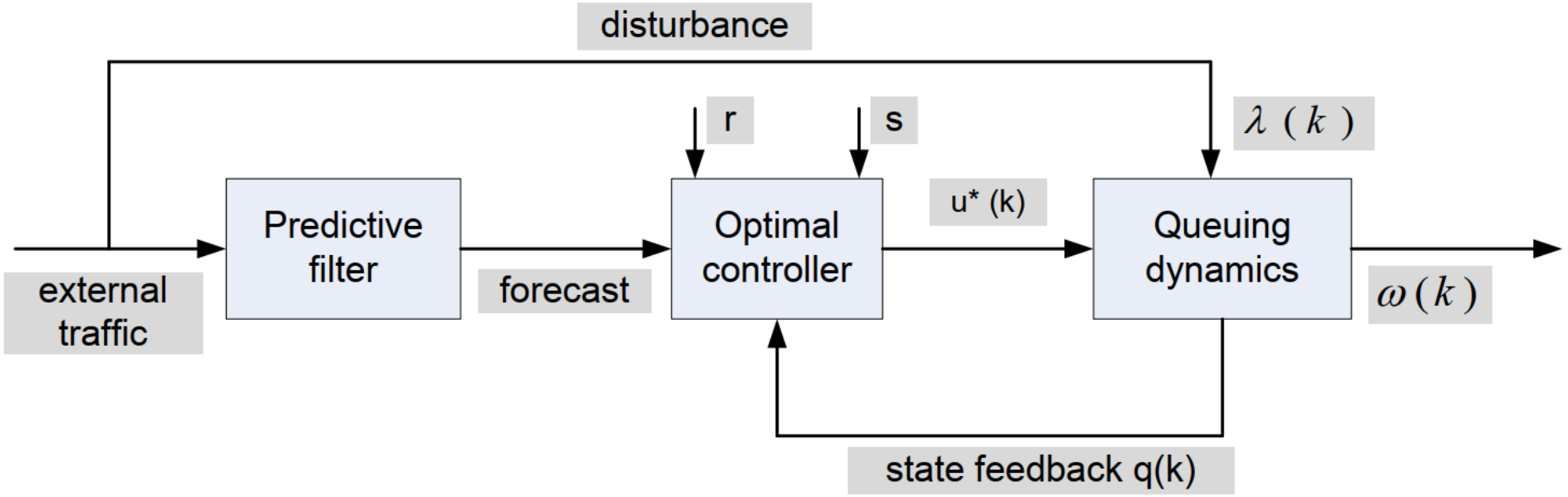
Komponen utama dari sistem kontrol:

- Input beban kerja → yang ditawarkan dan kebijakan untuk kontrol masuk, alokasi kapasitas, penyeimbangan beban, optimalisasi energi, dan jaminan QoS di cloud.
- Komponen sistem kontrol sensor → yang digunakan untuk memperkirakan ukuran kinerja yang relevan dan pengontrol yang menerapkan berbagai kebijakan.
- Output alokasi → sumber daya untuk masing-masing aplikasi.

Umpan Balik dan Stabilitas

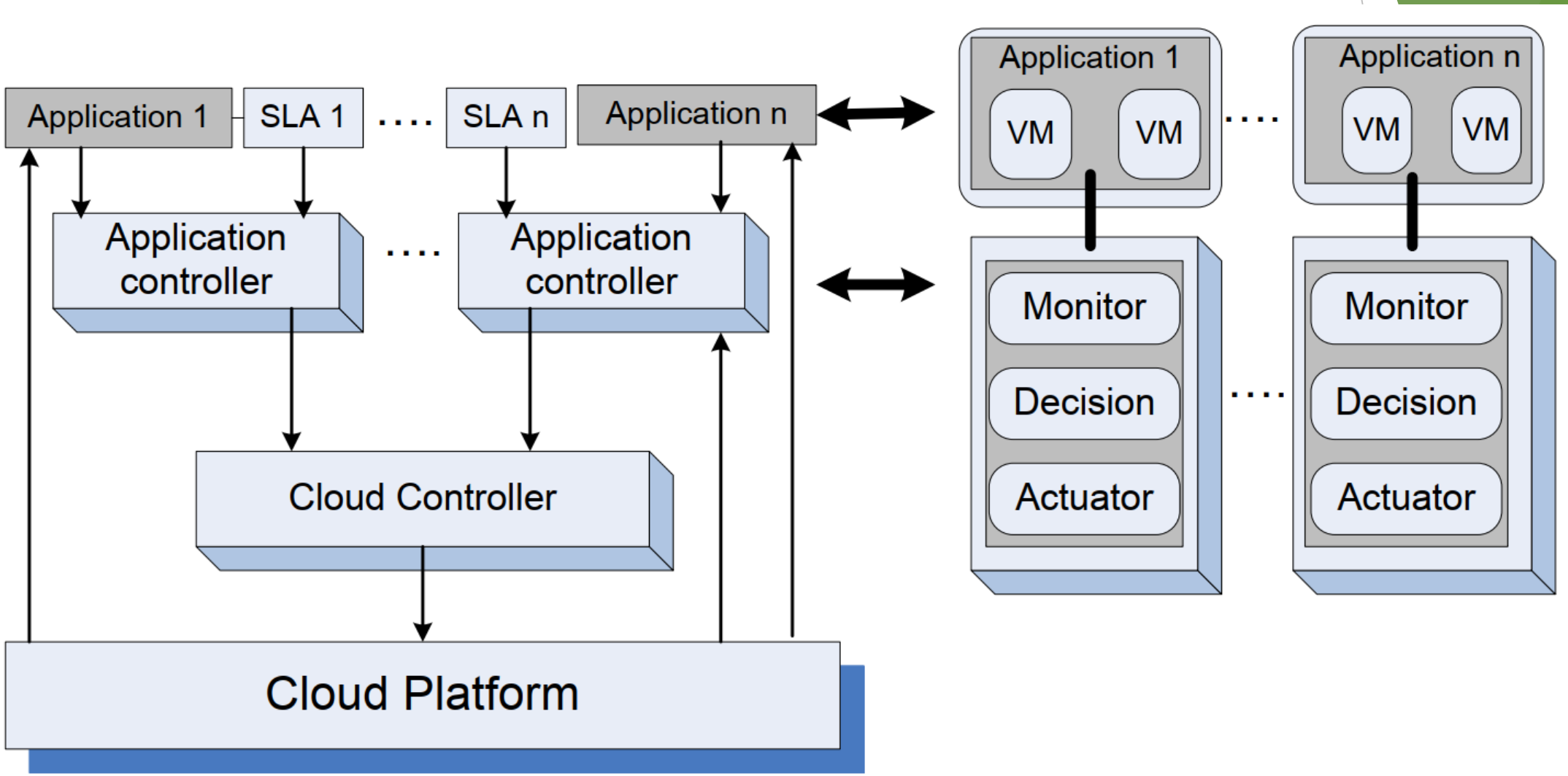
- Control granularity → tingkat detail informasi yang digunakan untuk mengontrol sistem.
  - Fine control → informasi yang sangat rinci tentang parameter yang mengontrol status sistem digunakan.
  - Coarse control → akurasi parameter ini ditukar dengan efisiensi implementasi.
- Controller → menggunakan umpan balik yang disediakan oleh sensor untuk menstabilkan sistem. Stabilitas terkait dengan perubahan output.
- Sumber ketidakstabilan dalam sistem kontrol apa pun:
  - Keterlambatan dalam mendapatkan reaksi sistem setelah tindakan kontrol.
  - The granularity of the control, fakta bahwa perubahan kecil yang dilakukan oleh pengontrol menyebabkan perubahan output yang sangat besar.
  - Osilasi, ketika perubahan input terlalu besar dan kontrol terlalu lemah, sehingga perubahan input merambat langsung ke output.

Struktur pengontrol cloud



Kontroler menggunakan umpan balik mengenai keadaan saat ini dan estimasi gangguan di masa depan karena lingkungan untuk menghitung input optimal pada prespetif yang terbatas. r dan s adalah faktor pembobotan indeks kinerja.

Pengontrol cloud dua tingkat



Pelajaran dari eksperimen dua tingkat

- Tindakan sistem kontrol harus dilakukan dalam ritme yang tidak mengarah pada ketidakstabilan.
- Penyesuaian hanya boleh dilakukan setelah kinerja sistem stabil.
- Jika ambang batas atas dan bawah ditetapkan, maka ketidakstabilan terjadi ketika mereka terlalu dekat satu sama lain jika variasi beban kerja cukup besar dan waktu yang dibutuhkan untuk beradaptasi tidak memungkinkan sistem untuk stabil.
- Tindakan terdiri dari alokasi/dealokasi satu atau lebih mesin virtual. Terkadang alokasi/dealokasi VM tunggal yang diperlukan oleh salah satu ambang batas dapat menyebabkan persilangan dengan yang lain, sumber ketidakstabilan lainnya.

Impementasi dan kosep kontrol poada CRM

- Mengatur parameter operasi utama sistem berdasarkan pengukuran keluaran sistem.
- Kontrol umpan balik mengasumsikan model sistem invarian waktu linier, dan pengontrol loop tertutup.
- Fungsi transfer sistem memenuhi batasan stabilitas dan sensitivitas.
- Sebuah ambang nilai parameter yang terkait dengan keadaan sistem yang memicu perubahan perilaku sistem.
- Ambang digunakan untuk menjaga parameter kritis dari suatu sistem dalam rentang yang telah ditentukan. Dua jenis kebijakan:
  1. berbasis ambang batas batas atas dan bawah pada adaptasi pemicu kinerja melalui realokasi sumber daya; kebijakan tersebut sederhana dan intuitif tetapi memerlukan pengaturan ambang batas per aplikasi.
  2. keputusan sekuensial berdasarkan model keputusan Markovian

Design decisions

- bermanfaat untuk memiliki dua jenis pengontrol:
  - pengontrol aplikasi menentukan apakah sumber daya tambahan diperlukan.
  - pengontrol Cloud menengahi permintaan sumber daya dan mengalokasikan sumber daya fisik.
- Pilih fine versus coarse control.
- Ambang dinamis berdasarkan rata-rata waktu lebih baik dibandingkan yang statis.
- Gunakan ambang batas tinggi dan rendah versus ambang batas tinggi saja.

Ambang batas proporsional

Algoritma

- Hitung nilai integral dari ambang batas tinggi dan rendah sebagai rata-rata maksimum dan, masing-masing, minimum pemanfaatan prosesor selama riwayat proses.
- Minta VM tambahan ketika nilai rata-rata penggunaan CPU selama irisan waktu saat ini melebihi ambang batas tinggi.
- Membebaskan VM ketika nilai rata-rata penggunaan CPU selama potongan waktu saat ambang batas rendah.

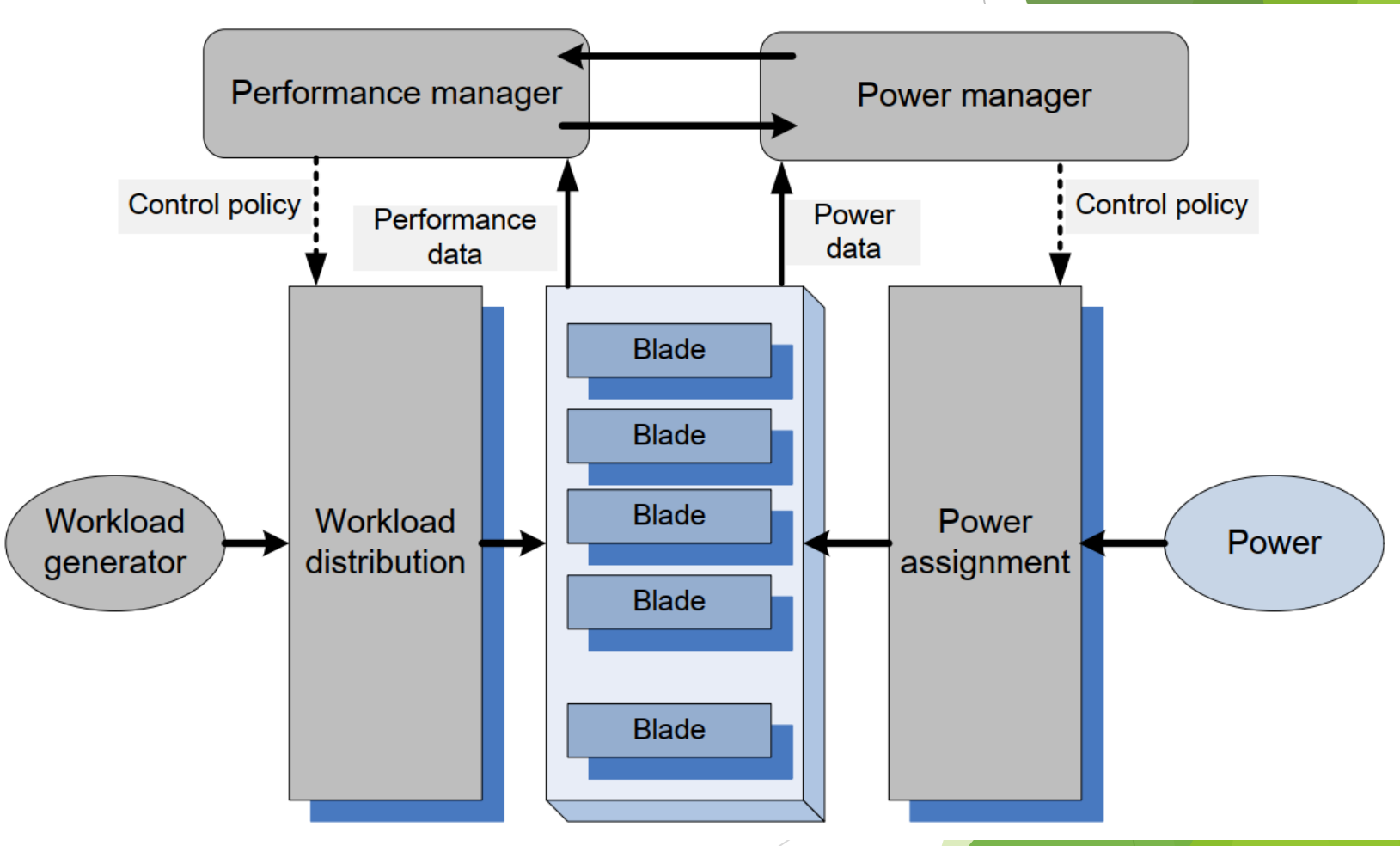
Kesimpulan Ambang dinamis berkinerja lebih baik daripada yang statis. Dua ambang lebih baik dari satu

Mengkoordinasikan daya dan manajemen kinerja

- Gunakan pengontrol terpisah untuk dua tujuan.
- Identifikasi satu set minimal parameter yang akan dipertukarkan antara dua pengontrol.
- Gunakan fungsi utilitas bersama untuk daya dan kinerja.
- Atur batas daya untuk sistem individual berdasarkan kebijakan manajemen daya yang dioptimalkan dengan utility.
- Gunakan manajer kinerja standar yang dimodifikasi hanya untuk menerima masukan dari pengontrol daya mengenai frekuensi yang ditentukan menurut kebijakan manajemen daya.
- Gunakan sistem perangkat lunak standar.

Komunikasi antara manajer otonom

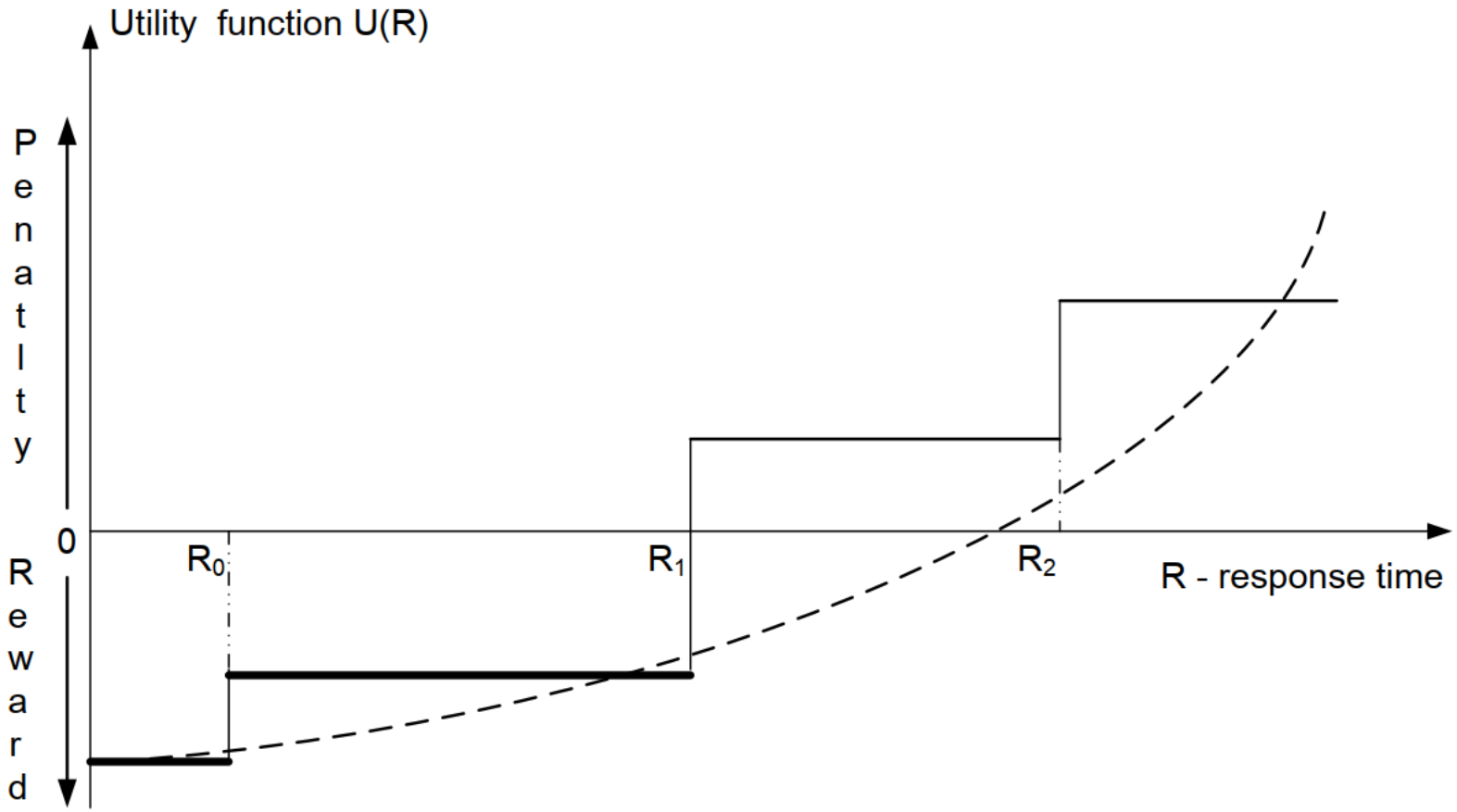
Kinerja otonom dan manajer daya bekerja sama untuk memastikan kinerja yang ditentukan dan optimalisasi energi; mereka diberi makan dengan data kinerja dan daya dan menerapkan kebijakan kinerja dan manajemen daya



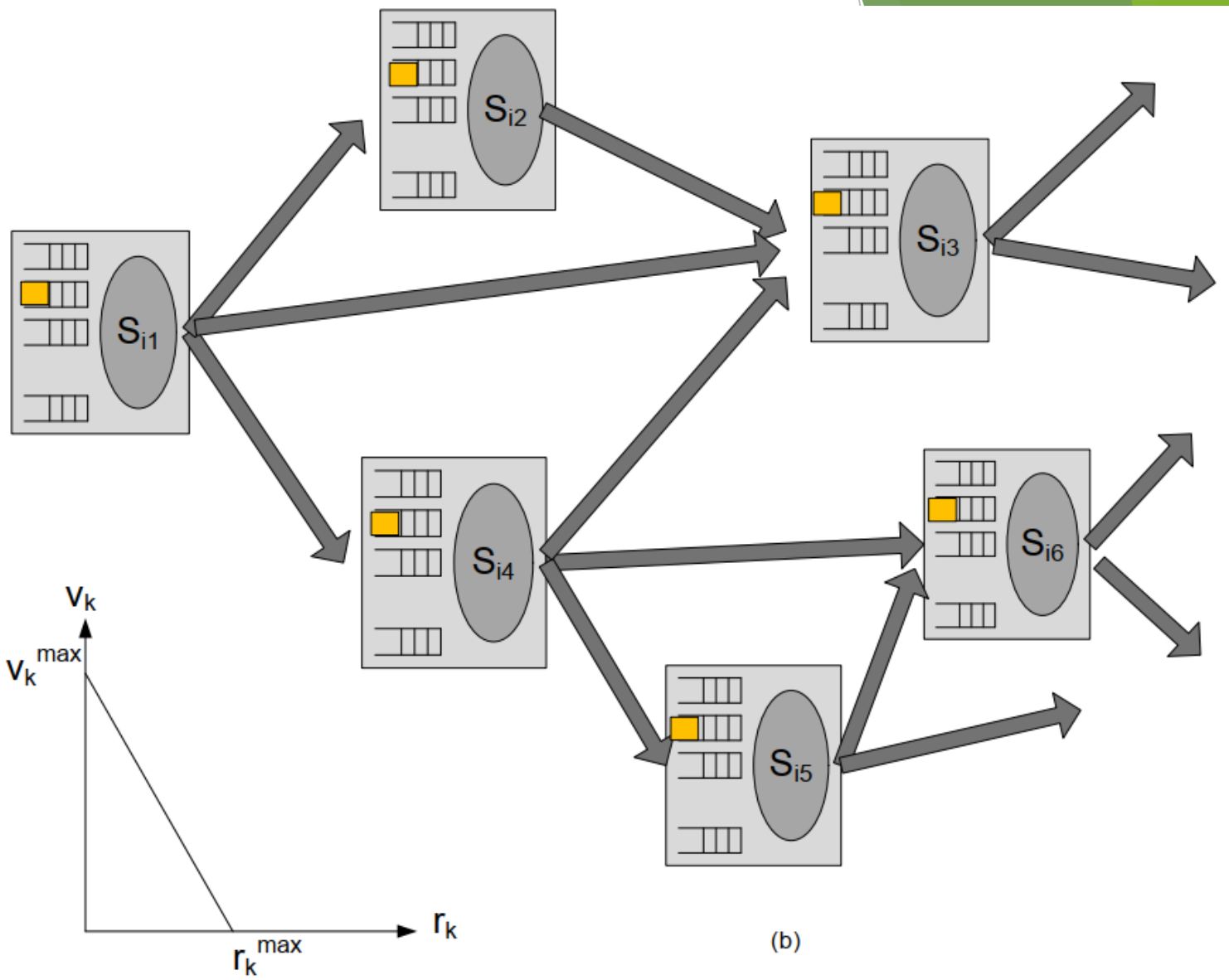
Model utilitas untuk layanan web berbasis cloud

- Sebuah perjanjian tingkat layanan (service level agreement (SLA)) → menentukan reward serta pinalti yang terkait dengan metrik kinerja tertentu.
- SLA untuk layanan web berbasis cloud menggunakan waktu respons rata-rata untuk mencerminkan Kualitas Layanan.
- Cloud menyediakan K kelas layanan yang berbeda, setiap kelas k melibatkan aplikasi N_K.
- Sistem dimodelkan sebagai jaringan antrian dengan banyak antrian untuk setiap server.
- Delay → memodelkan waktu berpikir pengguna setelah selesainya layanan di satu server dan dimulainya pemrosesan di server berikutnya.

Fungsi utilitas U(R) adalah serangkaian fungsi langkah dengan lompatan yang sesuai dengan waktu respons, R=R0 | R1 | R2, ketika level reward dan penalty berubah sesuai dengan SLA. Garis putus-putus menunjukkan pendekatan kuadrat dari fungsi utilitas.



(a) Fungsi utilitas: vk fungsi pendapatan (atau penalti) dari waktu respons rk untuk permintaan kelas k.
(b) Jaringan multiqueues



Model membutuhkan sejumlah besar parameter

Name	Description
I	the set of servers
K	the set of classes
Λ_k	the aggregate rate for class $k \in K$, $\Lambda_k = \lambda_k + \sum_{k' \in K} \Lambda_{k'} \pi_{k,k'}$
a_i	the availability of server $i \in I$
A_k	minimum level of availability for request class $k \in K$ specified by the SLA
m_k	the slope of the utility function for a class $k \in K$ application
N_k	number of applications in class $k \in K$
H_i	the range of frequencies of server $i \in I$
$C_{i,h}$	capacity of server $i \in I$ running at frequency $h \in H_i$
$c_{i,h}$	cost for server $i \in I$ running at frequency $h \in H_i$
$\bar{c}_i$	average cost of running server i
$\mu_{k,j}$	maximum service rate for a unit capacity server for tier j of a class k request
cm	the cost of moving a virtual machine from one server to another
cs_i	the cost for switching server i from the stand-by mode to an active state
$RAM_{k,j}$	the amount of main memory for tier j of class k request
$\overline{RAM_i}$	the amount of memory available on server i

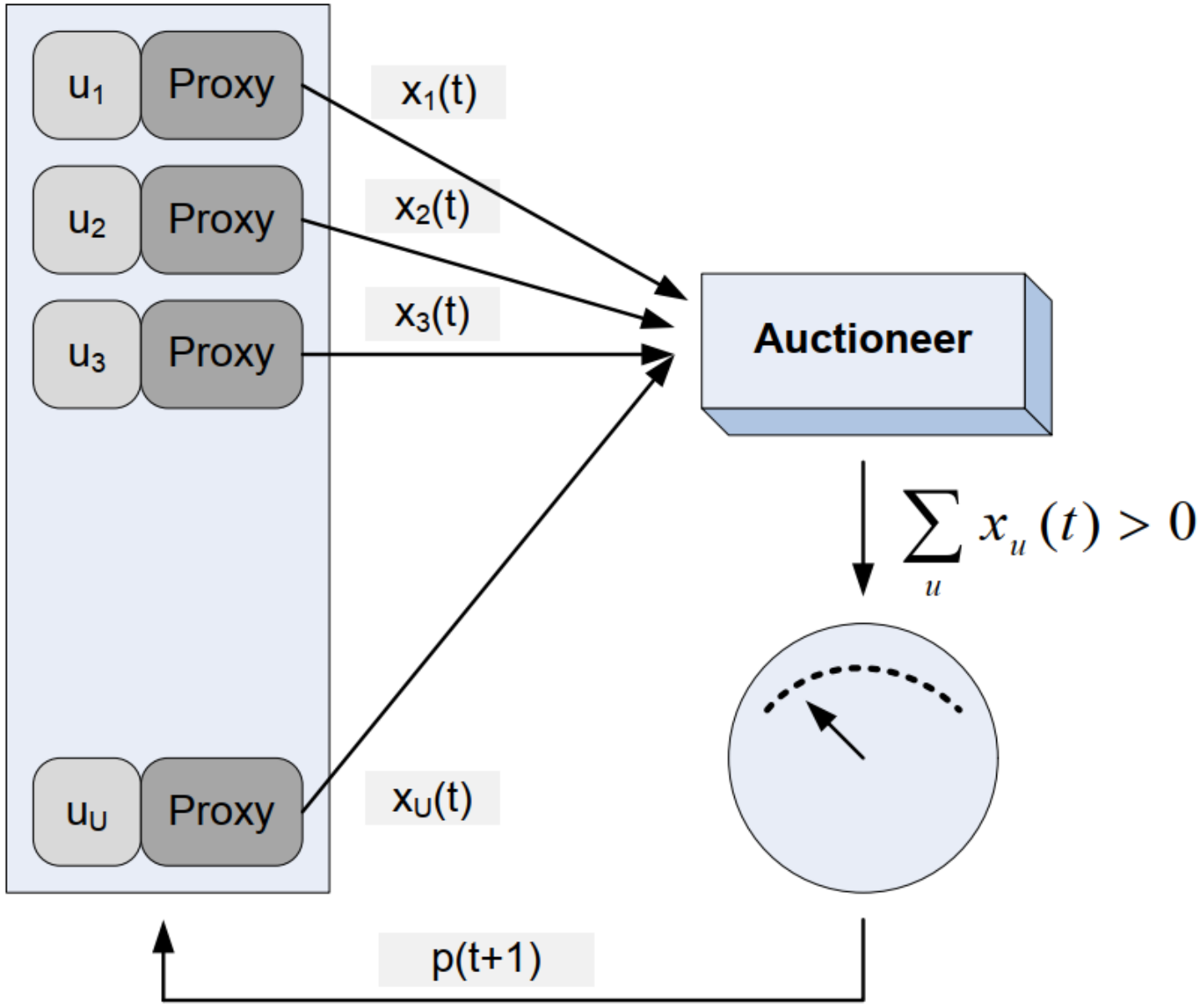
Pengelompokan sumber daya

- Sumber daya di cloud dialokasikan dalam kelompok.
- Pengguna mendapatkan manfaat maksimal dari kombinasi sumber daya tertentu: siklus CPU, memori utama, ruang disk, bandwidth jaringan, dan sebagainya.
- Penggabungan sumber daya memperumit model alokasi sumber daya.
- Proses bidding bertujuan untuk mengoptimalkan fungsi tujuan f(x,p).
- Dalam konteks komputasi awan, bidding adalah alokasi sumber daya kepada penawar tertinggi.

Kombinasi Sumber Daya cloud

- Pengguna memberikan tawaran untuk bundel yang diinginkan dan harga yang bersedia mereka bayar.
- Ascending Clock Auction, (ASCA) → harga saat ini untuk setiap sumber daya diwakili oleh "jam".
- Algoritme melibatkan penawaran pengguna dalam beberapa putaran; untuk mengatasi masalah ini, proxy pengguna secara otomatis menyesuaikan permintaan mereka atas nama penawar yang sebenarnya.

Skema algoritma ASCA; untuk memungkinkan pengguna lelang putaran tunggal diwakili oleh proxy yang menempatkan tawaran xu(t). Juru lelang menentukan apakah ada kelebihan permintaan dan, dalam hal itu, menaikkan harga sumber daya yang permintaannya melebihi penawaran dan meminta tawaran baru.



Algoritma penetapan harga dan alokasi

- Algoritme penetapan harga dan alokasi mempartisi kumpulan pengguna dalam dua kumpulan yang terpisah, pemenang dan kalah.
- Sifat yang diinginkan dari algoritme penetapan harga:
  - Mampu secara komputasional; algoritme lelang kombinatorial tradisional, misalnya, Vickey-Clarke-Groves (VLG) tidak dapat dilacak secara komputasi.
  - Skala dengan baik - mengingat skala sistem dan jumlah permintaan layanan, skalabilitas adalah kondisi yang diperlukan.
  - Bersikaplah objektif - pembagian pemenang dan pecundang hanya boleh didasarkan pada harga penawaran pengguna; jika harga melebihi ambang batas maka pengguna adalah pemenang, sebaliknya pengguna adalah pecundang.
  - Bersikap adil - pastikan harga seragam, semua pemenang dalam kumpulan sumber daya tertentu membayar harga yang sama.
  - Tunjukkan dengan jelas di akhir lelang harga satuan untuk setiap kumpulan sumber daya.Tunjukkan dengan jelas kepada semua peserta hubungan antara Penawaran dan permintaan dalam sistem.

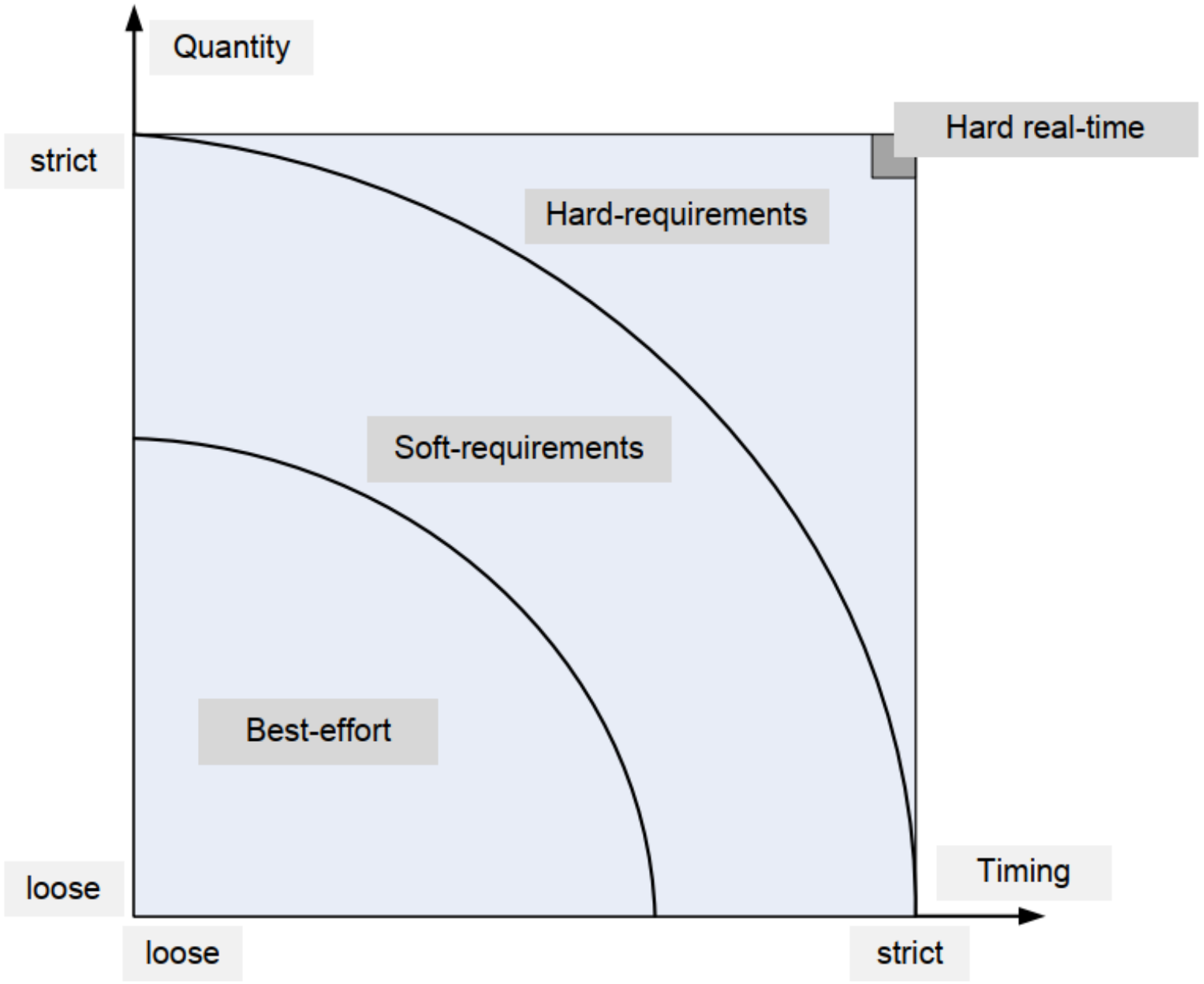
Algoritme penjadwalan cloud

- Penjadwalan → bertanggung jawab untuk berbagi sumber daya di beberapa tingkatan:
  - Sebuah server dapat dibagi di antara beberapa mesin virtual.
  - Mesin virtual dapat mendukung beberapa aplikasi.
  - Sebuah aplikasi dapat terdiri dari beberapa thread.
- Algoritma penjadwalan harus efisien, adil, dan starvation-free.
- Tujuan dari penjadwal:
  - Sistem batch → memaksimalkan throughput dan meminimalkan waktu penyelesaian.
  - Sistem waktu nyata(realtime) → memenuhi tenggat waktu dan dapat diprediksi.
- Upaya terbaik: aplikasi batch dan analitik.
- Algoritma umum untuk aplikasi upaya terbaik:
  - Round-robin.
  - First-Come-First-Serve (FCFS).
  - Shortest-Job-First (SJF).
  - Priority algorithms.

Algoritme penjadwalan cloud lanjutan

- Aplikasi multimedia (mis., streaming audio dan video)
  - Memiliki batasan waktu nyata.
  - Memerlukan delay dan throughput maksimum yang dijamin secara statistik.
- Aplikasi waktu nyata memiliki kendala waktu nyata yang sulit.
- Algoritma penjadwalan untuk aplikasi waktu nyata:
  - Batas Waktu Paling Awal Pertama (Earliest Deadline First EDF).
  - Tingkat Algoritma Monotonik (Rate Monotonic Algorithms RMA).
- Algoritma untuk penjadwalan terintegrasi dari beberapa kelas aplikasi:
- Alokasi/Pengiriman Sumber Daya (Resource Allocation/Dispatching RAD).
- Batas Waktu Awal Berbasis Tarif (Rate-Based Earliest Deadline RBED).

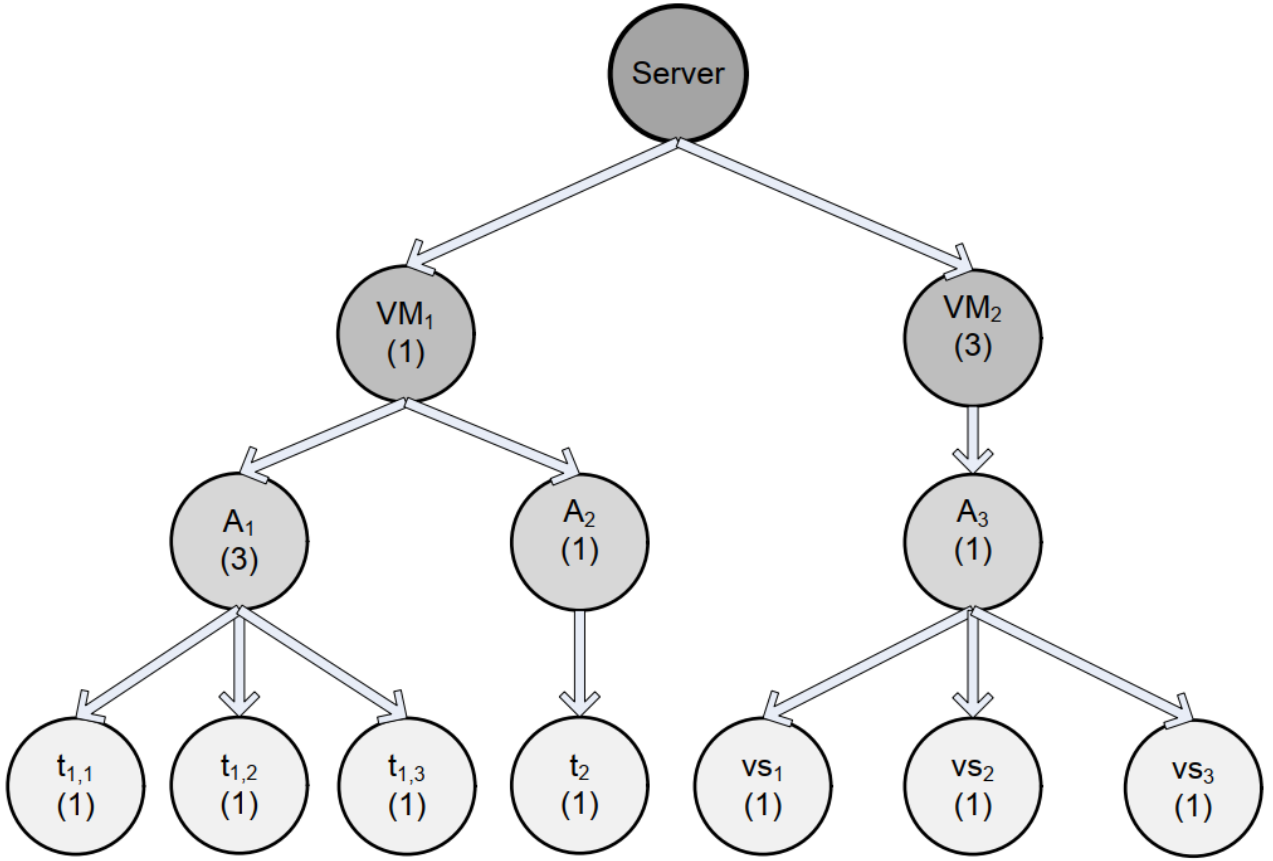
- Best-effort policies → tidak memaksakan persyaratan baik mengenai jumlahsumber daya yang dialokasikan untuk aplikasi, atau waktu ketika aplikasi dijadwalkan.
- Soft-requirements policies → memerlukan jumlah dan batasan waktu yang dijamin secara statistik.
- Hard-requirements policies → menuntut waktu yang ketat dan jumlah sumber daya yang tepat.



Waktu Mulai dan antrian yang adil

- Atur konsumen bandwidth CPU dalam struktur pohon.
- Node akar adalah prosesor dan daun dari pohon ini adalah utas dari setiap aplikasi.
  - Ketika mesin virtual tidak aktif, bandwidthnya dialokasikan kembali ke VM lain yang aktif pada saat itu.
  - Ketika salah satu aplikasi mesin virtual tidak aktif, alokasinya ditransfer ke aplikasi lain yang berjalan pada VM yang sama.
  - Jika salah satu utas aplikasi tidak dapat dijalankan maka alokasinya ditransfer ke utas aplikasi lainnya.

Pohon SFQ untuk penjadwalan ketika dua mesin virtual VM1 dan VM2 berjalan di server.



Penjadwalan cloud

- Hard deadlines → jika tugas tidak diselesaikan sebelum batas waktu, tugas lain yang bergantung padanya mungkin terpengaruh dan ada hukuman; tenggat waktu yang ketat sangat ketat dan dinyatakan dengan tepat sebagai milidetik, atau mungkin detik.
- Soft deadlines → lebih merupakan pedoman dan, secara umum, tidak ada hukuman; tenggat waktu lunak dapat dilewatkan dengan pecahan unit yang digunakan untuk mengekspresikannya, misalnya menit jika tenggat waktu dinyatakan dalam jam, atau jam jika tenggat waktu dinyatakan dalam hari.
- Hanya mempertimbangkan tugas aperiodik dengan beban kerja yang dapat dibagi secara acak.

Aturan pembagian beban kerja

- Optimal Partitioning Rule (OPR) beban kerja dipartisi untuk memastikan waktu penyelesaian sedini mungkin dan semua tugas harus diselesaikan pada waktu yang sama.
- Equal Partitioning Rule (EPR) menetapkan beban kerja yang sama ke node pekerja individu

Link Video penjelasan

https://drive.google.com/file/d/1K25oRd19o-KFzQi74UJ7U-0JFDNEYyy5/view?usp=sharing

Terimakasih