POWER & THROUGHPUT ISSUES IN
NEXT-GENERATION PACKET SWITCHES

Nick Bambos
Stanford University
The Rising Problem of Power (Density)

Switching: chips ~100W+ ... systems ~ KW ... clusters/farms/centers ~ 10KW++

Computing: processors ~100W+/− ... systems ~200W+

Racks: ~ 512 CPUs+ ... ... ... ~100KW+ ... **high power density** ... *liquid cooled!*

Data centers: 100,000 CPUs+ ... ... ... ~10MW++ ... high power density

High power distribution cost... high power bills... **high cooling costs**...
Electronic infrastructure costs get amortized ... power costs don’t...

Both matter: power *volume* and power *density* (perhaps more)
High/uneven power density ... high thermal stress ... fast component aging ...

**reliability problems**...
Throughput/speed can potentially... be adjusted by

- frequency scaling... clocking up/down
- voltage scaling...
- component shutdown (path/structure scaling)
- ...

Power scales up super-linearly with throughput/speed
Where/How to Attack the Problem

Level 3: algorithms and operations

Level 2: system architecture

Level 1: low-power circuit design
The Basic Model ...
A Canonical Model of Interacting Resource Allocation

X = backlog vector
S = service vector/mode/configuration/allocation
S = set of all possible service vectors
\( P_S \) = cost (power...) of service configuration \( S \)

**Core Issue**... dynamically choose \( S \) (based on backlog/service history, etc.)

to maximize throughput, minimize (power/delay) cost...
2x2 switch … simplest model (…not simplistic)… scales to NxN

Service vectors \( Sa \) and \( Sb \)
Multiprocessors ...

Allocating 3 processors to two job queues

\[ X_1 \rightarrow \begin{array}{c}
\text{fast} \\
\text{normal} \\
\text{slow}
\end{array} \]

... in general ... *any set* of service vectors
Throughput ...
Load and Throughput...

Arbitrary traffic traces

**Load** \( \rho = (\rho_1, \rho_2, \ldots, \rho_q, \ldots, \rho_Q) \) ... long term avg. packet load

\[ \rho_q = \frac{\text{packet load arriving to queue q in (0,t)}}{t} \] ... as t grows large

**Throughput** .... job departure rate = job arrival rate

... in/out flow balance

... **flow conservation**

\[ \frac{X(t)}{t} \rightarrow 0 \] ... at large times t
Throughput ... Capacity Region

\[ R = \{ \rho : \rho \leq \sum_{S \in S} \phi_S \text{ S... for some } \phi_S > 0 \text{ with } \sum \phi_S = 1 \} \]
Projective Cone Schedules (PCS) Maximize Throughput

**PCS algorithm**... when backlog \( X \), choose \( S \) to maximize projection on \( BX \)

\[
\max <S,BX> \quad \text{over } S \text{ in } S
\]

maximizes throughput for *any* fixed matrix \( B \) that is **positive-definite, symmetric** and has **negative/zero off-diagonal** elements

**MWM algorithm** ... PCS with \( B=I \)

Rich family of schedules... ( \( \sim Q^2 \) matrix parameters to tweak and tune)

Extremely **robust** schedules

Interesting geometric intuition
Geometry of Projective Cone Schedules (PCS)

Given $X$, choose $S$ to maximize $<S, BX>$ over all $S$ in $S$.

When $X$ in cone $C$,
choose $S = S(C)$ corresponding to that cone.
A 3-Queue Example

\[ S_1=(9,0,0) / S_2=(0,8,0) / S_3=(0,0,8) / S_4=(3,4,3) \]

\[ B=[1,0,0; 0,1,0; 0,0,1] \]

\[ B=[1,0,0; 0,2,0; 0,0,1] \]

\[ B=[1,-0.5,0; -0.5,1,0; 0,0,1] \]

\[ B=[1,-0.5,0; 0,1,0; 0,0,1] \]
Assume bound on “workload displacement/jump”

Have to search only neighbor cones ... fewer as workload grows! ... **Local Search**
When at $S$, check out $<S,BX>$ on **neighbor nodes** (only) ...

... chose $S'$ with maximal $<S,BX>$ ... **steepest ascent**
Service = Departures - Arrivals
Power ...
The Canonical Model Again... Power vs. Delay

\[ \{X_1, X_2, ..., X_{q}, X_Q\} \]

\[ \{S_1, S_2, ..., S_{q}, S_Q\} \]

- \(X = \) backlog vector
- \(S = \) service vector/mode/configuration/allocation
- \(S = \) set of all possible service vectors
- \(P_S = \) cost (power...) of service configuration \(S\)

**Core Issue...** dynamically choose \(S\) to **minimize power + delay cost**

Dynamic programming formulation ... intractable...
Activating faster/slower, higher/lower-power, more/less expensive service vectors

...adjusting the capacity space
Packet Switches ... with Deceleration Modes

Speed Mode $m$: slower/cooler ... faster/hotter
Service Vector $S = \text{Speed Mode } m \times \text{Port-Matching Configuration } C$

In each slot, have to choose $m$ & $C \ldots S = mC$

Backlog Cost... $X'EX$ ...per slot ... $E$ pos-def & sym

Power Cost ... $S'FS$ ...per slot ... $F$ pos-def & sym

No arrivals...

$$J(X) = \min[ X'EX + S'FS - J(X-S/\text{depts}) ] \quad \text{min over } S$$

Continuous Relaxation ... **Linear Quadratic Regulator (LQR)... Ricatti Equation**

$$S^*(X) = \left[ (F+P)^{-1} \ P \right] X$$

Where $P$ solves $E = P (F+P)^{-1} P$
Power-Aware MWM... PA-MWM

Reduced Backlog Cost... $X'EX \sim b \text{ Sum}[X_q^2]$  
Reduced Power Cost ... $S'FS \sim f [\text{ Sum } S_q]^2$

**PA-MWM algorithm**

Choose Configuration $C^*(X) = \text{argmax } <C,X>$

Choose Speed Mode $m^*(X) = \text{argmin } |m - a \text{ TotalBacklog}|$
Total Cost = BacklogCost + a PowerCost … vary ‘a’
Thank You!