

Workshop on the Frontiers in Distributed Communication, Sensing and Control
October 31 - November 2, 2008
Yale University

Message Passing Algorithms for Stochastic Loss Networks

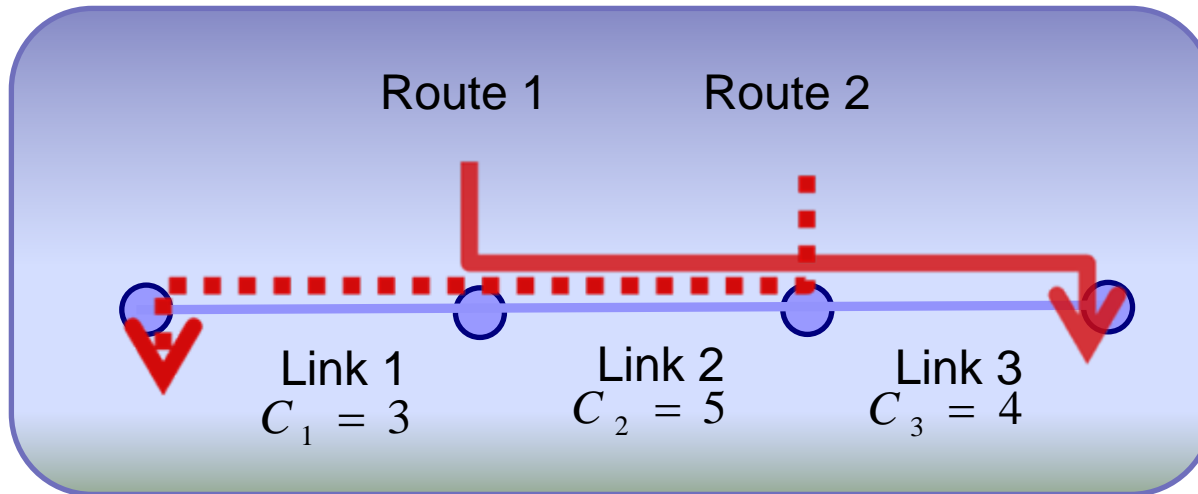
Mayank Sharma

Joint work with:

Kyomin Jung & Devavrat Shah (MIT)
Yingdong Lu & Mark Squillante (IBM T.J. Watson Research Center)

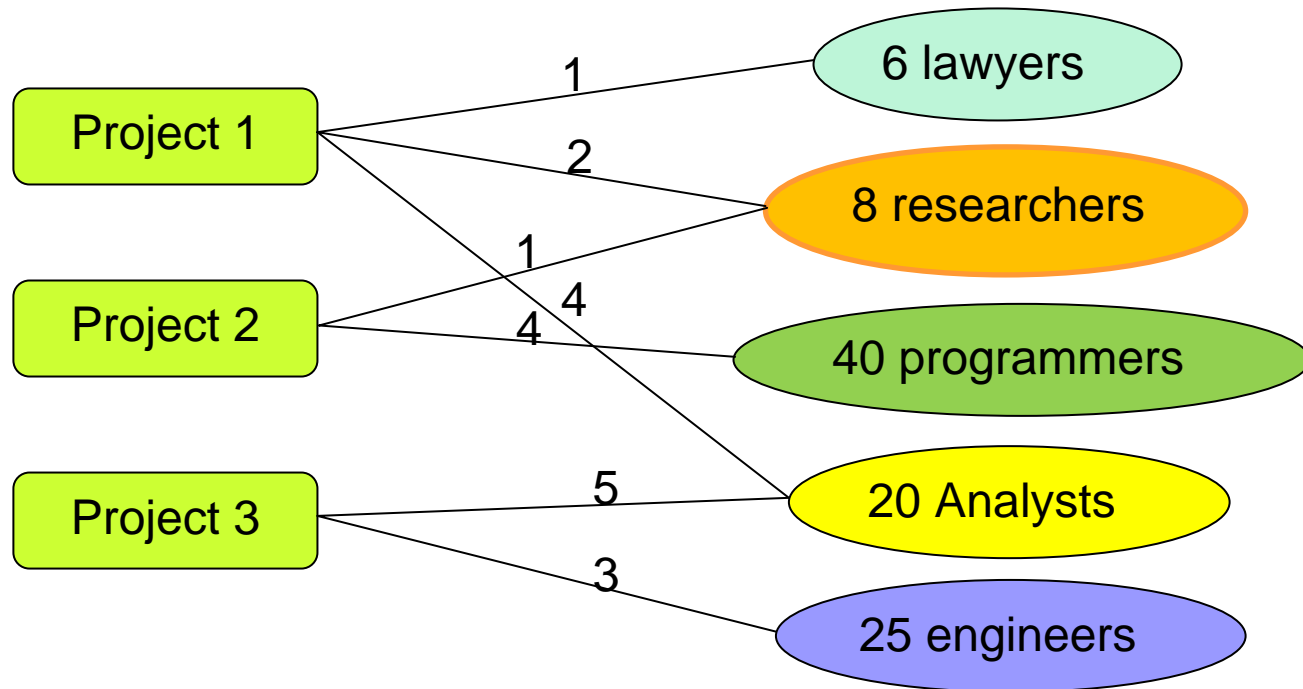
(presented at ACM SIGMETRICS 2008)

Loss network



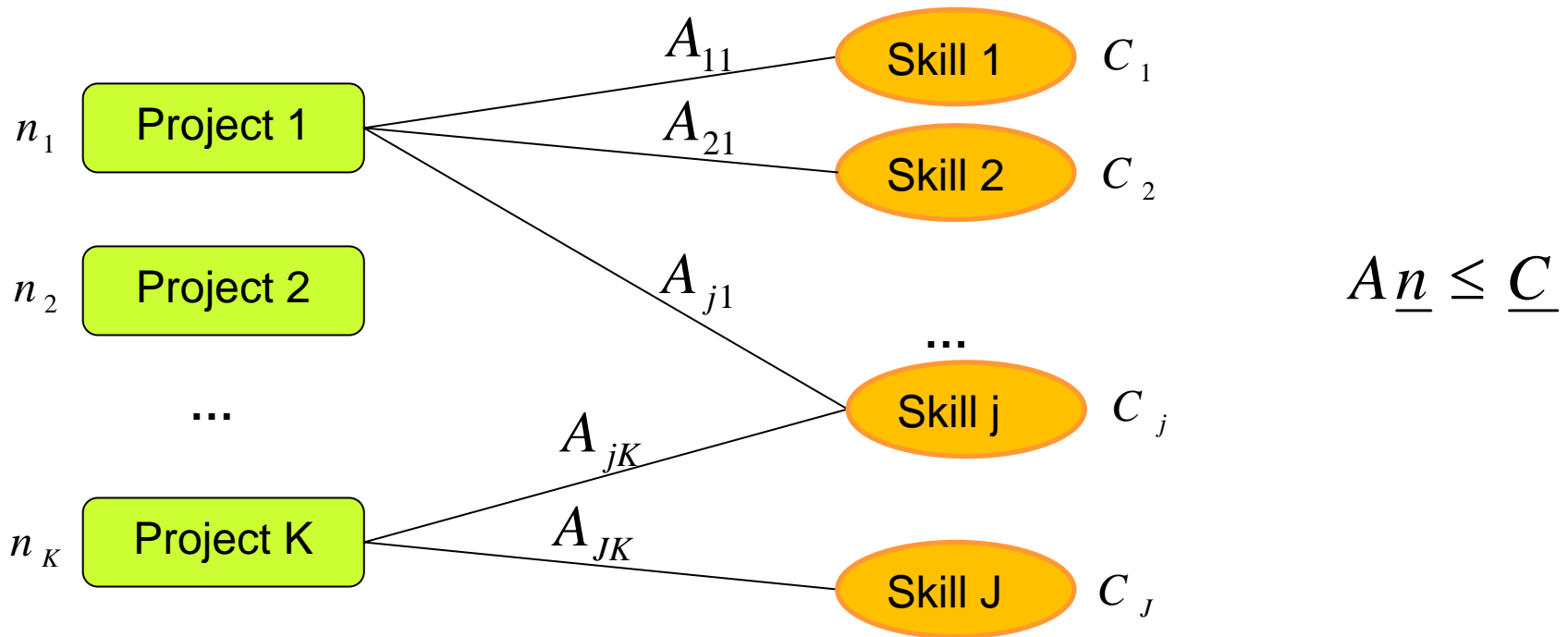
- Given a set of links with capacity constraints and a set of routes that support calls
 - A call is dropped if any of the links on its route is full.
- Applications include classical telephony, circuit-switched networks, ATM networks, etc.
- Models the problem of Simultaneous Resource Possession
 - A route may consist of any subset of the links.

Resource allocation problem



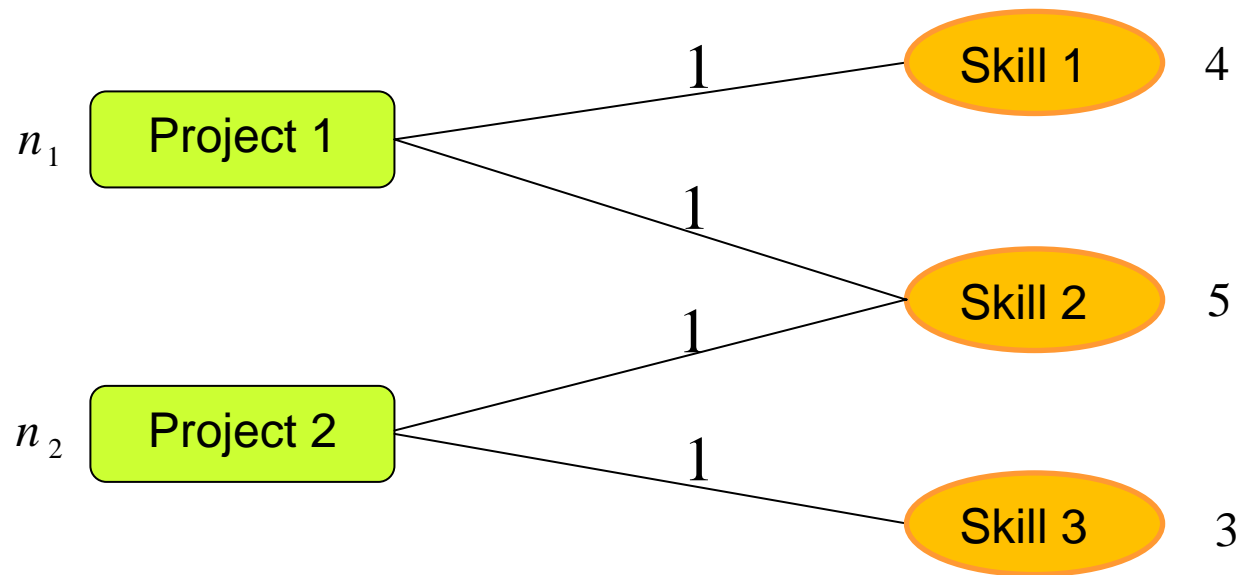
- Problem: manage the workforce given staffing constraints and stochastic demand processes for projects with diverse staffing requirements.

Resource allocation problem



- often non-integer A matrix and capacity constraints C
- possibly non-singular A matrix

Resource allocation problem



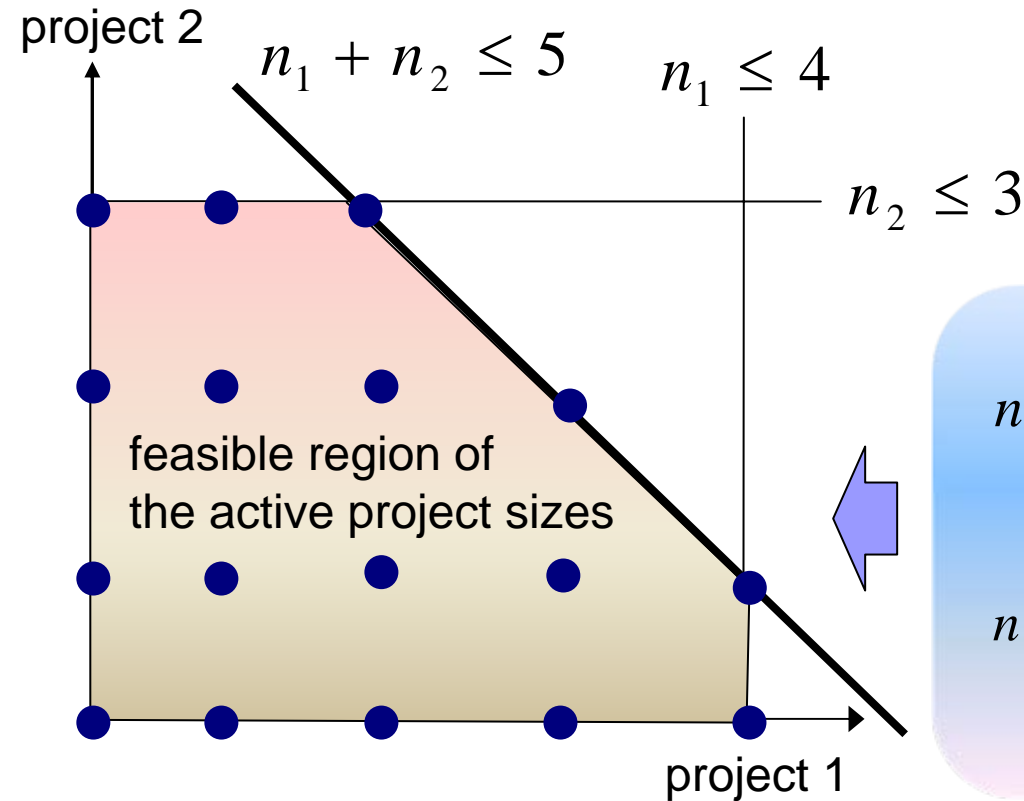
$$\underline{C} = [4, 5, 3]^T$$

$$\underline{n} = [n_1, n_2]^T$$

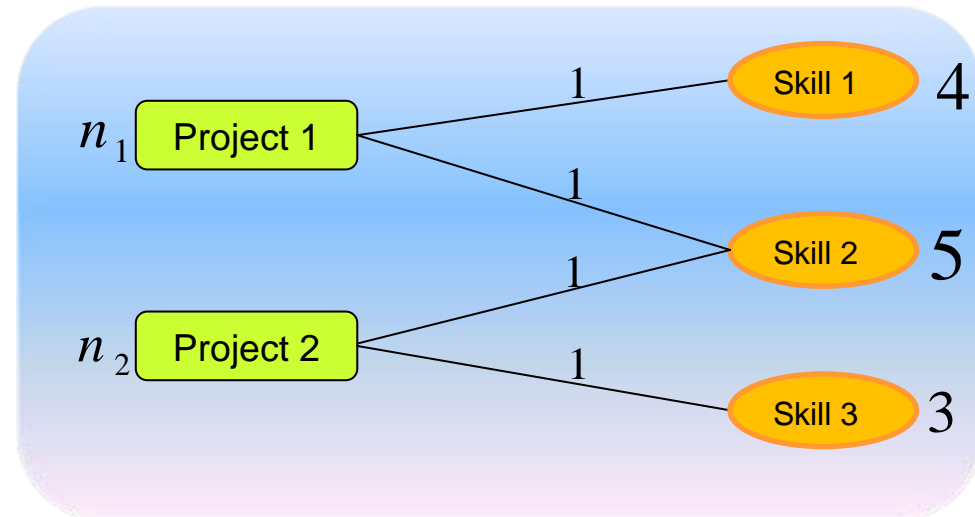
$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$A\underline{n} \leq \underline{C}$$

Feasible Region

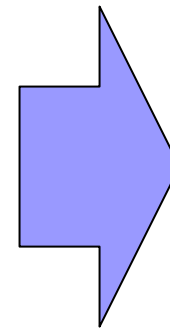
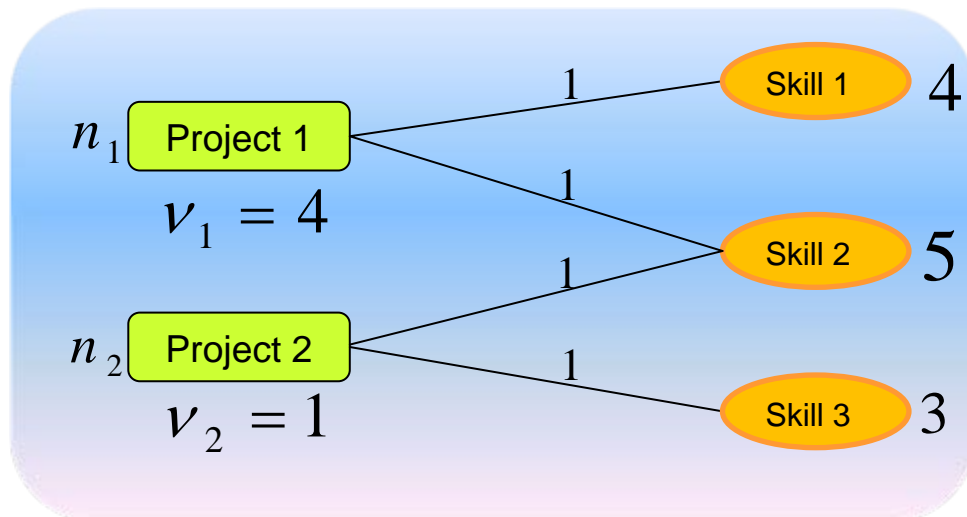


$$A \underline{n} \leq \underline{C}$$



Stochastic loss network

- We use the classical loss network model to study the problem.
- Project requests arrive according to i.i.d. Poisson processes.
- Let $\underline{v} = (v_1, \dots, v_K)$ be the vector of arrival rates
 - unit service rate



$$\underline{v} = (4, 1)$$

$$\underline{C} = (4, 5, 3)$$

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}$$

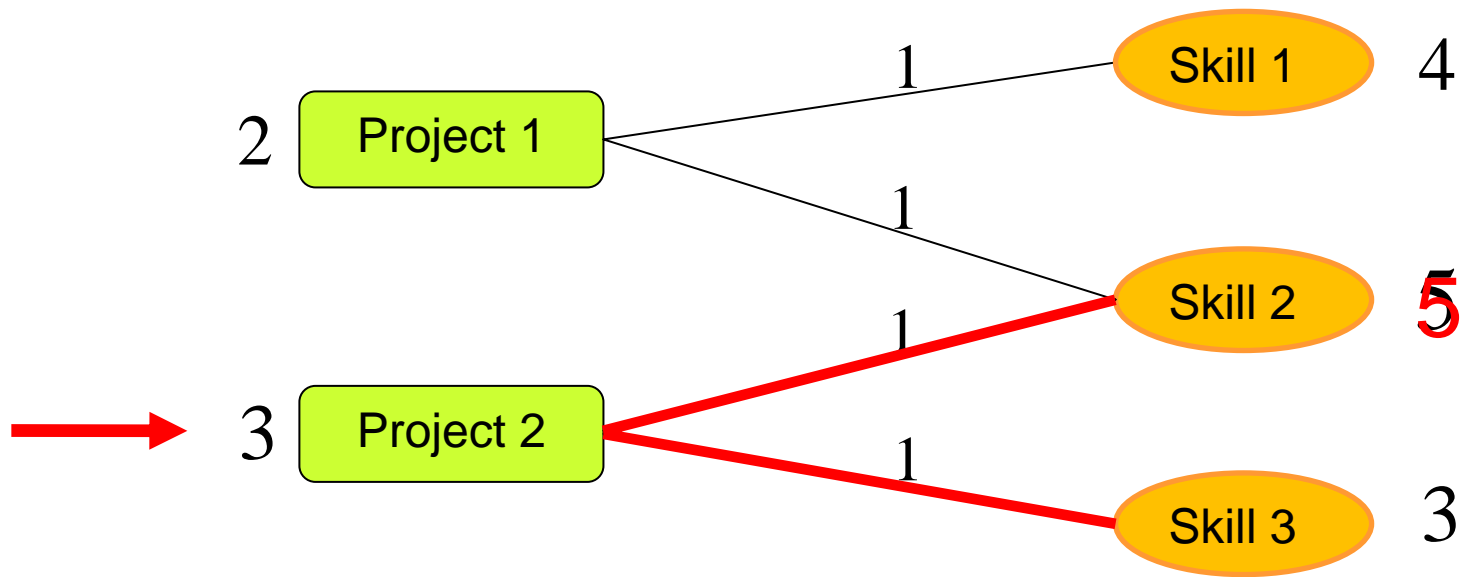
Stochastic loss network

- Let $\underline{n}(t) = (n_1(t), \dots, n_K(t)) \in Z_+^K$ be the vector of the number of active projects at time t .
- We have that $\underline{n}(t) \in S(\underline{C}) = \{\underline{n} \in Z_+^K : A\underline{n} \leq \underline{C}\}$ where $S(\underline{C})$ is the feasible region of the state space.
- $\underline{n}(t)$ forms a reversible Markov chain
 - It has a product form stationary distribution (cf. Kelly '79)

$$\pi(\underline{n}) = G(\underline{C})^{-1} \prod_{r \in R} (v_r^{n_r} / n_r!)$$

for $\underline{n} \in S(\underline{C})$, where $G(\underline{C}) = \sum_{\underline{n} \in S(\underline{C})} \prod_{r \in R} (v_r^{n_r} / n_r!)$

Loss probability of projects



$$\underline{A}n \leq \underline{C}$$

Loss probability of projects

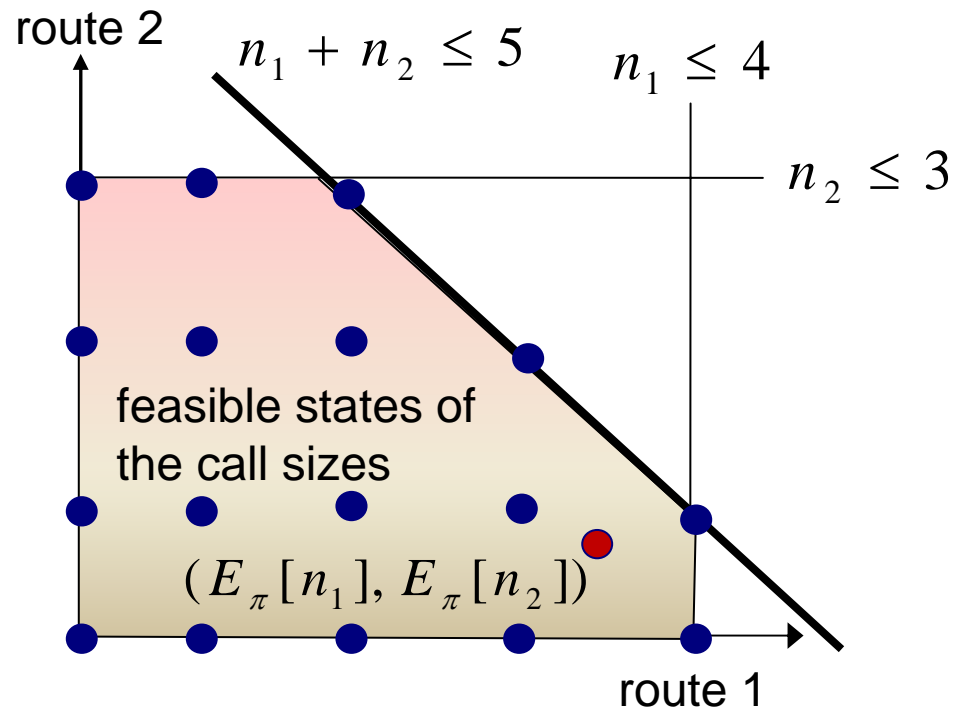
- A primary performance measure is the per-project stationary loss probability L_r ,
 - the fraction of requests for project-type r that are dropped.

$$1 - L_r = \sum_{n \in S(C - Ae_r)} \pi(n) = G(\underline{C})^{-1} G(\underline{C} - Ae_r)$$

- By Little's law, we have $E[n_r] = v_r (1 - L_r)$

Loss probability of projects

- Computing the vector $(E[n_1], E[n_2], \dots, E[n_K])$ is equivalent to computing loss probabilities.



- Exact computation of loss probability is #P-complete [Louth, Mitzenmacher, Kelly '94].

Message-passing paradigm

- Given a probability distribution, compute efficiently:
 - arbitrary marginals
 - mode of probability distribution
 - or approximations thereof.
- If probability distribution is derived from a Graphical Model
 - Belief propagation (BP): sum-product, max-product, min-sum
 - exciting new developments from statistical physics community
- Recent work on using BP to calculate loss probabilities
 - Ni & Tatikonda [2005, 2007]

Scaling of the network

- We consider the following scaling of the large network first introduced by F. Kelly'86.

- Let N be the system scaling parameter.

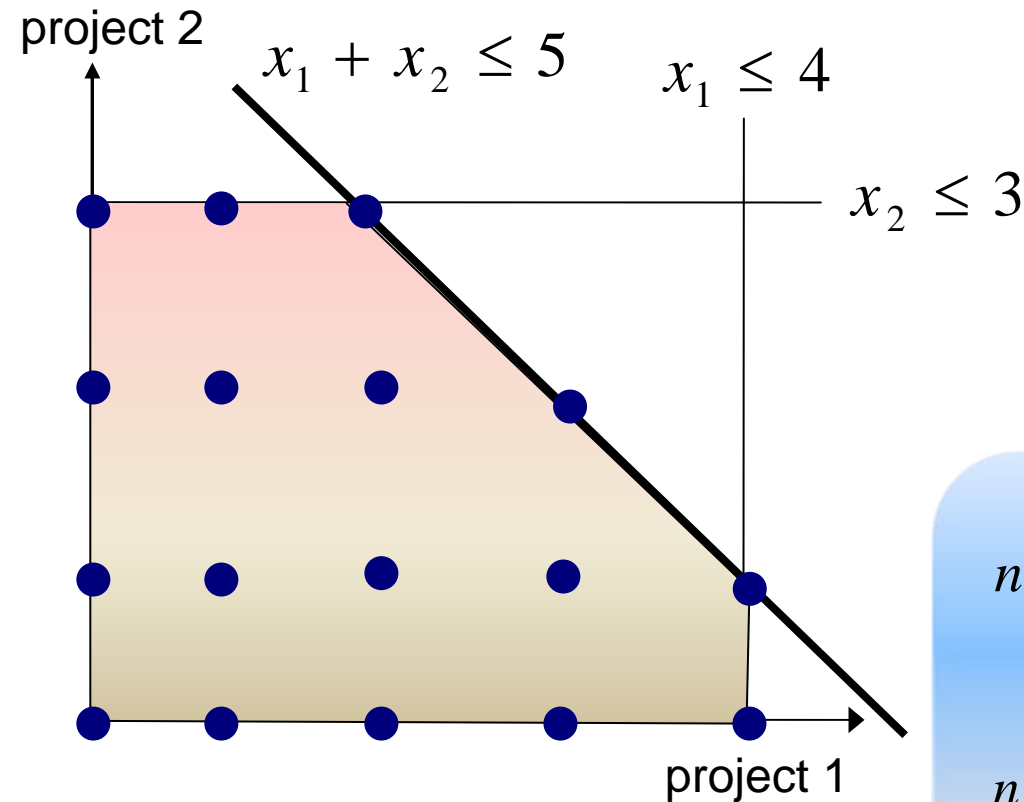
- Then $A_{\underline{n}} \leq \underline{NC}$, $\underline{n} \in Z_+^K$ or equivalently

- $A_{\underline{n}} \leq \underline{C}$, $\underline{n} \in \{0, \frac{1}{N}, \frac{2}{N}, \frac{3}{N}, \dots\}^K$

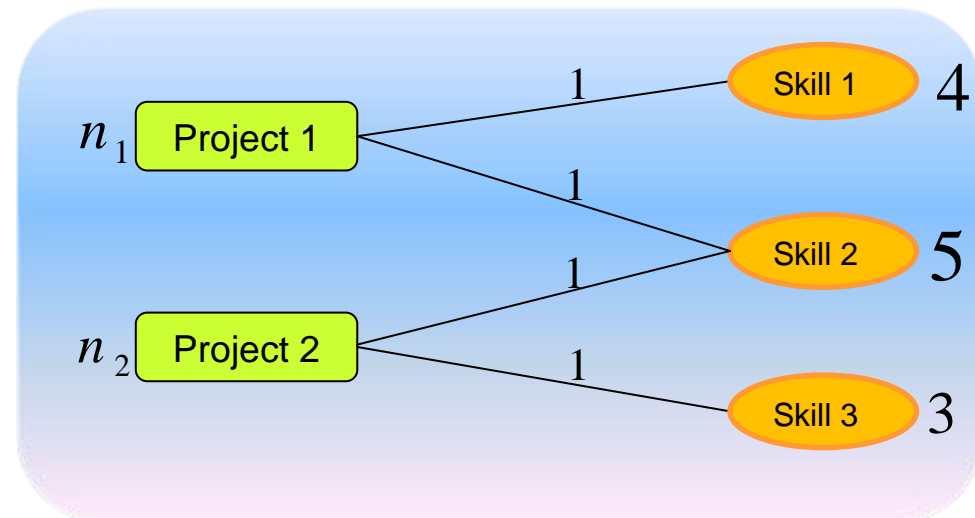
- The stationary distribution for N scale system :

$$\pi_N(\underline{n}) \propto \prod_{r \in R} \frac{(N \nu_r)^{N n_r}}{(N n_r)!}$$

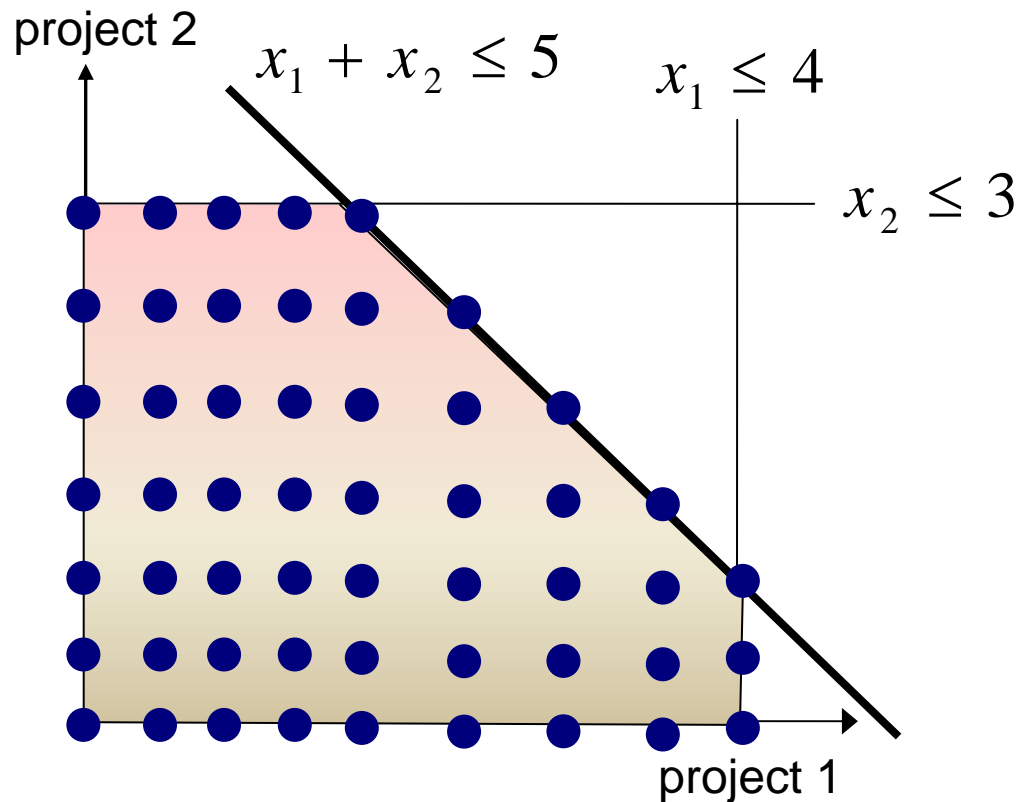
Scaling of the network



$$N = 1$$

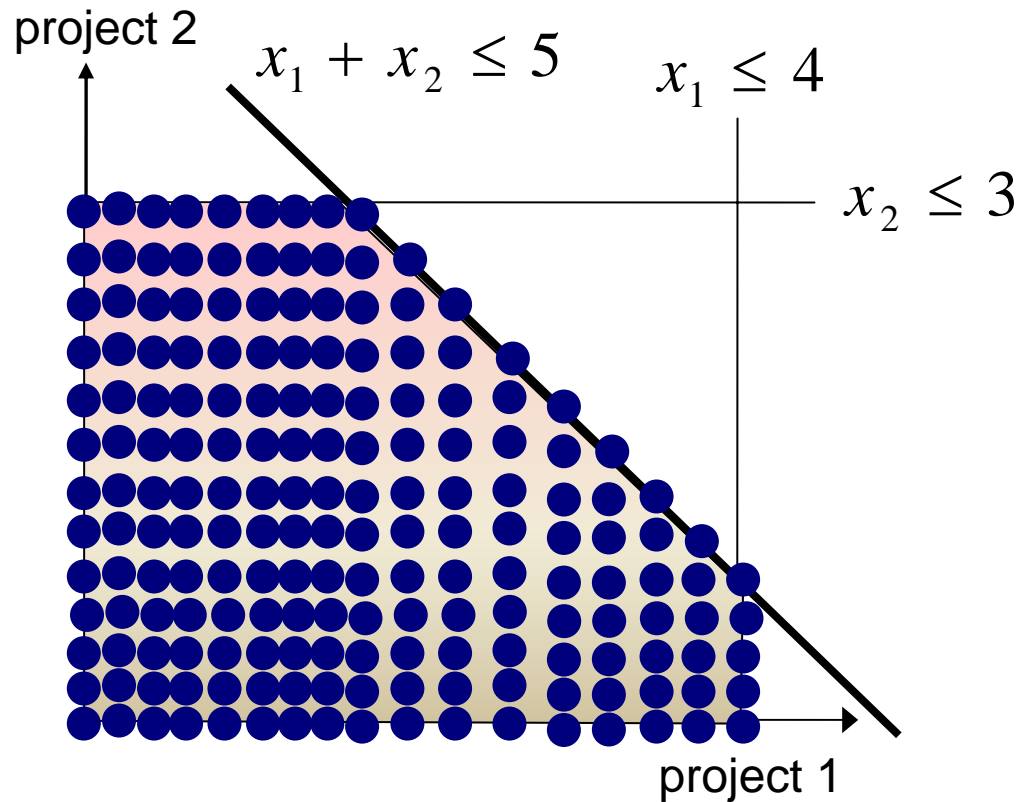


Scaling of the network



$$N = 2$$

Scaling of the network



$$N = 4$$

- As N increases, the constraints become $A\underline{x} \leq \underline{C}, \underline{x} \in R_+^K$

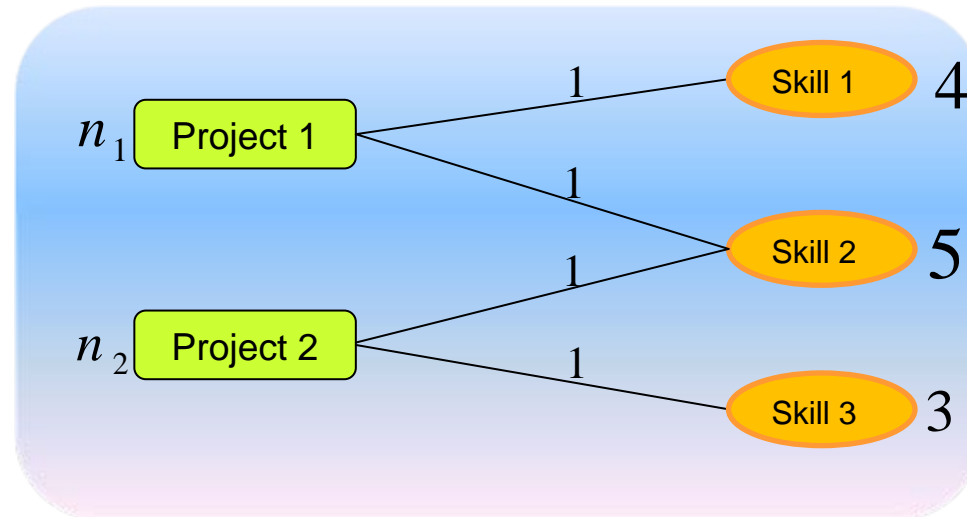
Erlang Fixed Point approximation

- Classical approach for this problem is the Erlang Fixed Point approximation (EFP).
- EFP is based on the well known Erlang formula [E'48] for blocking (loss) probability for an $M/G/1/K$ queue with capacity K and arrival rate ρ :

$$E(\rho, K) = \frac{\frac{\rho^K}{K!}}{\frac{\rho^0}{0!} + \frac{\rho^1}{1!} + \frac{\rho^2}{2!} + \dots + \frac{\rho^K}{K!}}$$

Erlang Fixed Point approximation

- Main intuition of EFP is from the hypothesis that project requests are lost due to independent blocking events on each skills for the project.



$$1 - L_2 = (1 - B_2)(1 - B_3), \quad B_j: \text{blocking probability of skill } j.$$

$$B_2 = E(\nu_1(1 - B_1) + \nu_2(1 - B_3), C_2), \quad B_3 = E(\nu_2(1 - B_2), C_3)$$

Erlang Fixed Point approximation

- Formally, EFP computes loss probabilities of routes $\underline{L} = (L_1, \dots, L_K)$ and blocking probabilities of links $\underline{B} = (B_1, \dots, B_J)$ that satisfies the fixed point equations :

$$B_j = E(\rho_j, C_j),$$
$$\rho_j = \frac{1}{1-B_j} \left[\sum_r v_r A_{jr} \prod_i (1-B_i)^{A_{ir}} \right],$$
$$1 - L_r = \prod_j (1 - B_j)^{A_{jr}},$$

for $j=1,2,\dots,J$ and $r \in R$.

EFP: message-passing for scaled system

- **PRIMAL:** find mode $\underline{n}^* = \arg \max_{\underline{n} \in S(\underline{C})} \pi(\underline{n})$

$$\max \sum_{r \in R} n_r \log v_r - \log n_r!$$

subject to $\underline{n} \in S(\underline{C})$.

- **DUAL:**

$$\min \sum_{r \in R} v_r \exp \left[- \sum_j y_j A_{jr} \right] + \sum_j y_j C_j$$

subject to $\underline{y} \geq 0$.

Erlang Fixed Point approximation

- Theorem (EFP error upper bound)

- Let L_r^N be the exact loss probability of project r , and $L_r^{E,N}$ be the loss probability computed from EFP. Then

$$|L_r^{E,N} - L_r^N| = O\left(\sqrt{\frac{\log N}{N}}\right)$$

(established by Kelly in 1986).

- We reprove this result using variational characterization of the stationary distribution, which yields a simpler set of arguments.
- When the arrival rate vector \underline{v} lies strictly inside or strictly outside of the feasible region, it can be checked that the error in the EFP is $O\left(\frac{1}{N}\right)$.

Erlang Fixed Point approximation

- Theorem (EFP error lower bound)

- When the arrival rate vector \underline{v} lies on the boundary of the feasible region,

$$\| L^{E,N} - L^N \|_2 = \Omega\left(\frac{1}{\sqrt{N}}\right)$$

where $L^N = (L_r^N)$ is the vector of exact loss probability and $L^{E,N} = (L_r^{E,N})$ is the vector of the loss probability from EFP.

- First result showing error lower bound of EFP.

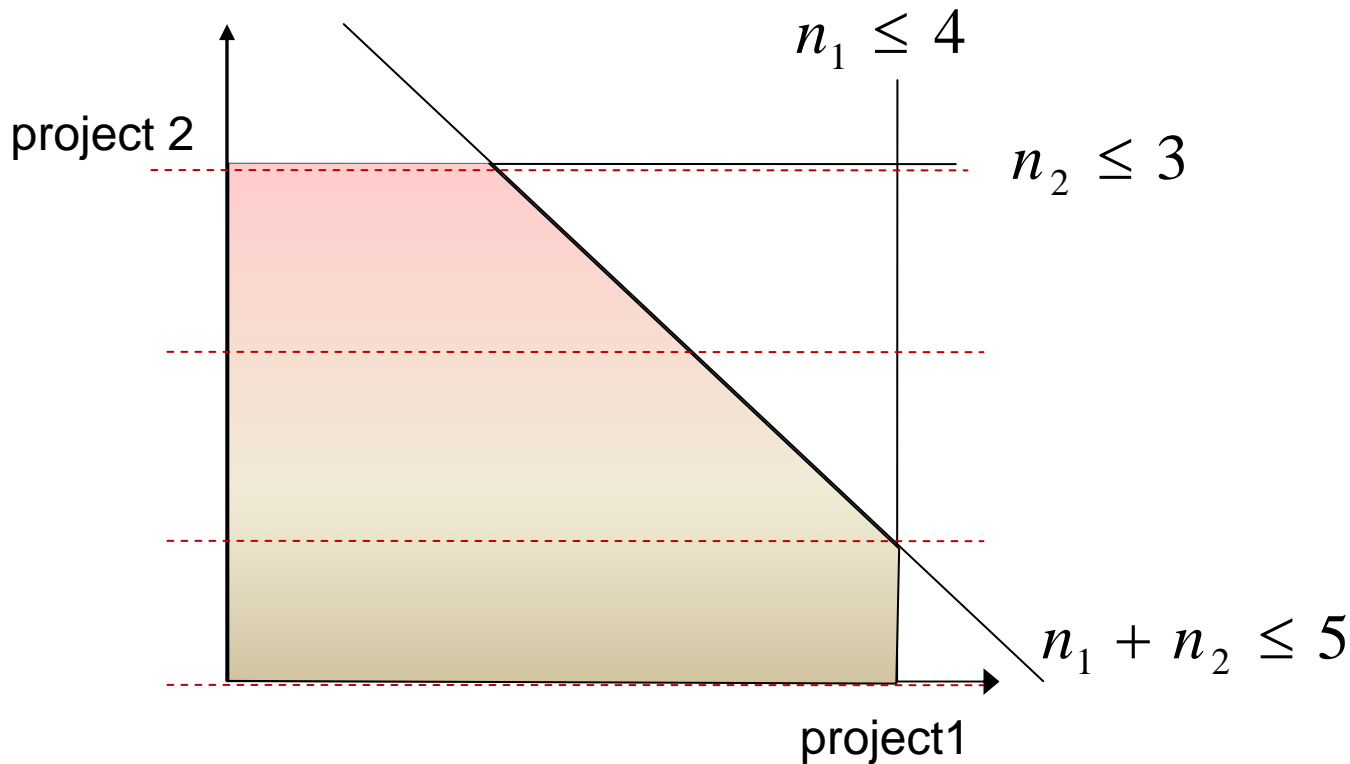
- Main ideas of the proof are :

- When the arrival rate vector lies on the boundary, symmetry of probability distribution does not hold.
- Use spherical integration.

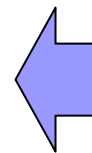
New Algorithm : Slice method

- In a nut-shell, Erlang approximation computes
 - $(E[n_1], E[n_2], \dots, E[n_K]) \approx n^*$ where
 - n^* is the most probable state (mode of the distribution).
- This works well when arrival rate vector is away from the boundary
 - But, when the rate vector lies on the boundary,
 - Due to asymmetry, it works poorly.
- Hence, we need a better algorithm
 - That goes beyond this approximation by mode

New Algorithm : Slice method

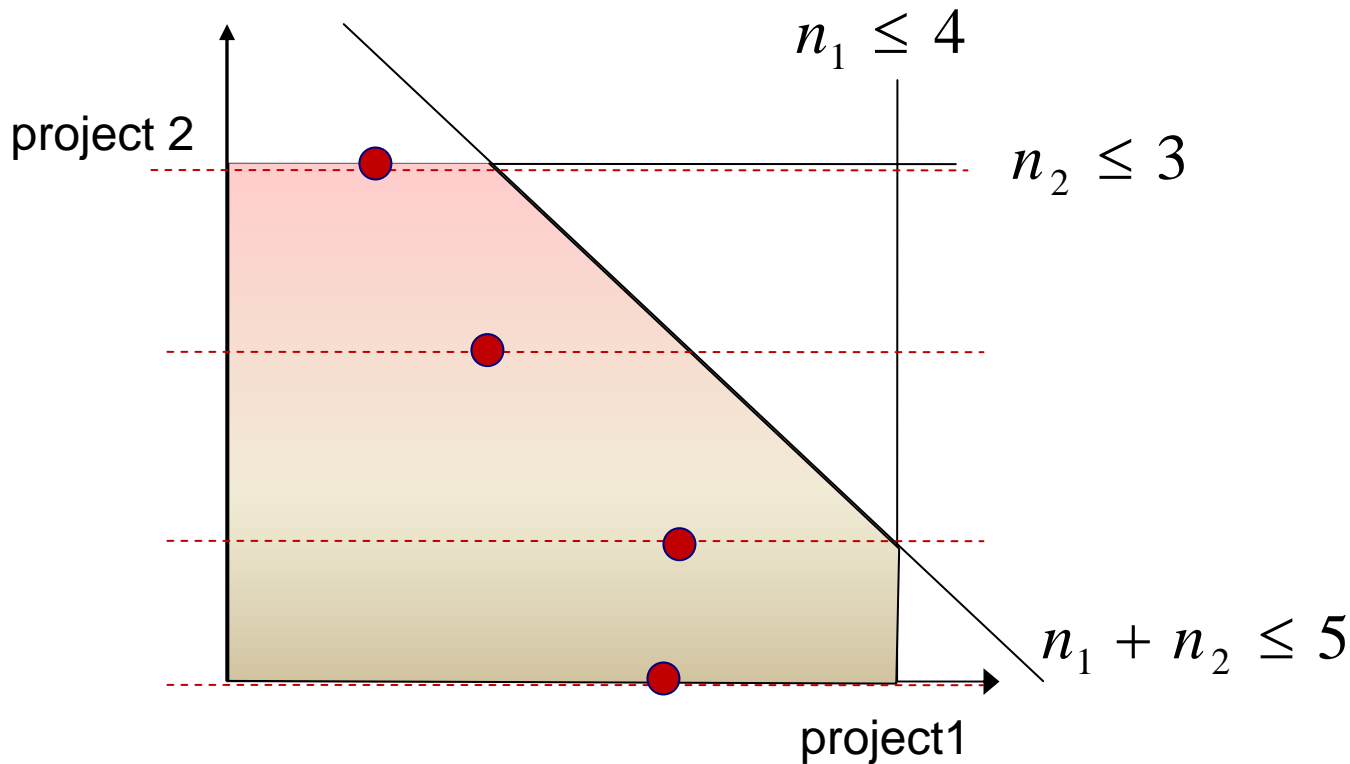


$$E[n_2] = \sum_{k=0}^{\infty} kP[n_2 = k]$$



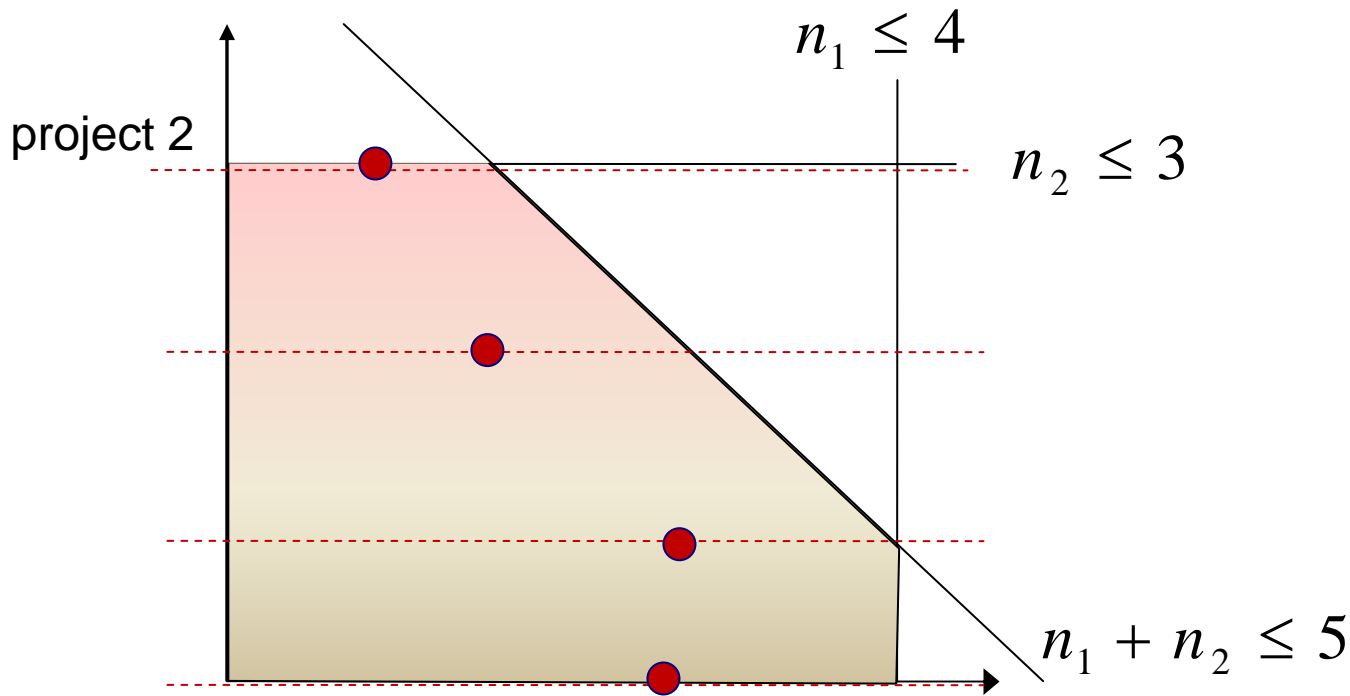
Approximate $P[n_2 = k]$.

New Algorithm : Slice method



$$P[n_2 = k] \approx P[n_2 = k; n_1 = n_{1k}^*] \leftarrow \text{Approximate } P[n_2 = k].$$

New Algorithm : Slice method



$$E^*[n_2] = \frac{\sum_{k=0}^{\infty} k P[n_2 = k; n_1 = n_{1k}^*]}{\sum_{k=0}^{\infty} P[n_2 = k; n_1 = n_{1k}^*]}$$

Slice method

- Theorem (Slice method error upper bound)

- Let L_r^N be the exact loss probability of skill r , and $L_r^{S,N}$ be the loss probability computed from the Slice method.

$$|L_r^{S,N} - L_r^N| = O\left(\sqrt{\frac{\log N}{N}}\right)$$

- We design an iterative message-passing algorithm implementation
 - It has an exponential convergence (or linear rate of convergence)
- Similar to the bound on the Erlang approximation.
- Numerical results illustrate that the slice method can convincingly outperform the EFP under critical load.

Message passing implementation

- To compute the approximate solution of n_k^* on each slice, we compute approximate solution z_j^* for its dual.

$$n_k^* = \nu_r \prod_j (z_j^*)^{A_{jr}}$$

- We provide iterative, coordinate descent algorithm for computation of the dual solution.
- The algorithm computes the dual solution by passing messages through the edges of the factor graph of the loss network.

Message passing implementation

- Initialize $z_j^{(0)} = 0.5$ for all $1 \leq j \leq J$
- Update in round-robin fashion by solving fixed-point equations:

$$g_j^{(t+1)}(x) = \min \{C_j, g_j^{(t)}(1)\}$$

where

$$g_j^{(t+1)}(x) = \sum_{r \in R} \nu_r A_{jr} \prod_i z_i^{A_{jr}}$$

$$\text{with } z_i = \begin{cases} z_i^{(t+1)} & i < j \\ x & i = j \\ z_i^{(t)} & i > j \end{cases}$$

Numerical results

■ Small network example

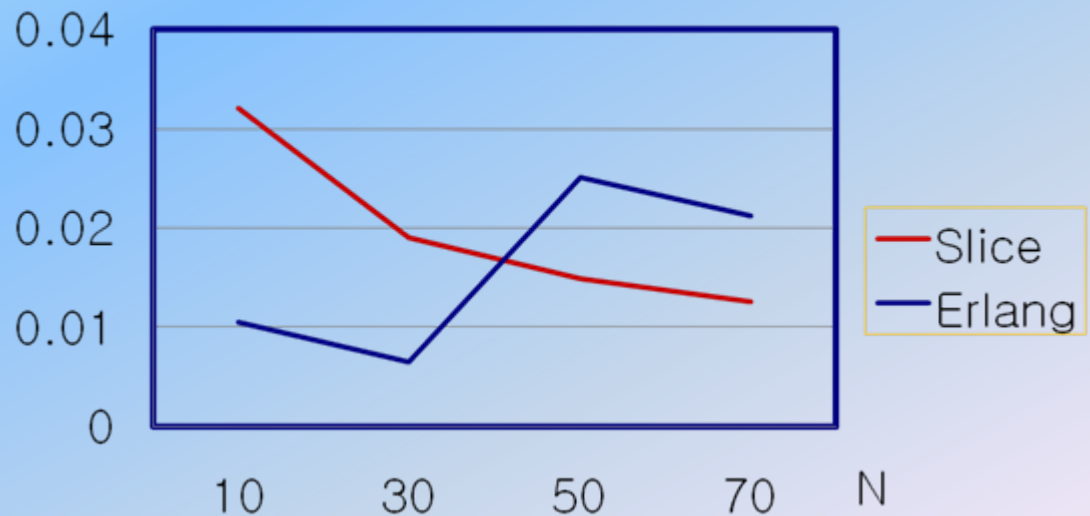
$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\underline{C} = (2, 3, 2)$$

$$\underline{v} = (2, 1)$$



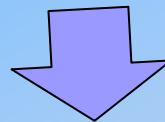
Error Average loss prob. error



Numerical results

■ Large real-world application

- Instances from resource planning applications in the workforce management in the IT service industry.
- First model has 38 routes and 132 links, and the second model has 110 routes and 132 links.



Average loss prob. error

	Erlang	Slice
Model 1	0.3357	0.1720
Model 2	0.3847	0.0923

Conclusion and Future Work

- Erlang approximation
 - Works well when network is under or over loaded.
 - We find it works poorly when the network is critically loaded.
- We proposed a new algorithm
 - Achieves good approximation for the critically loaded case.
 - Message-passing implementation converges exponentially fast.
- Future direction
 - Better message-passing algorithms

Thank you.