Blocking Probabilities of Two-Layer Statistically Indistinguishable Multicast Streams

Jouni Karvo, Samuli Aalto & Jorma Virtamo
Networking Laboratory
Helsinki University of Technology

samuli.aalto@hut.fi

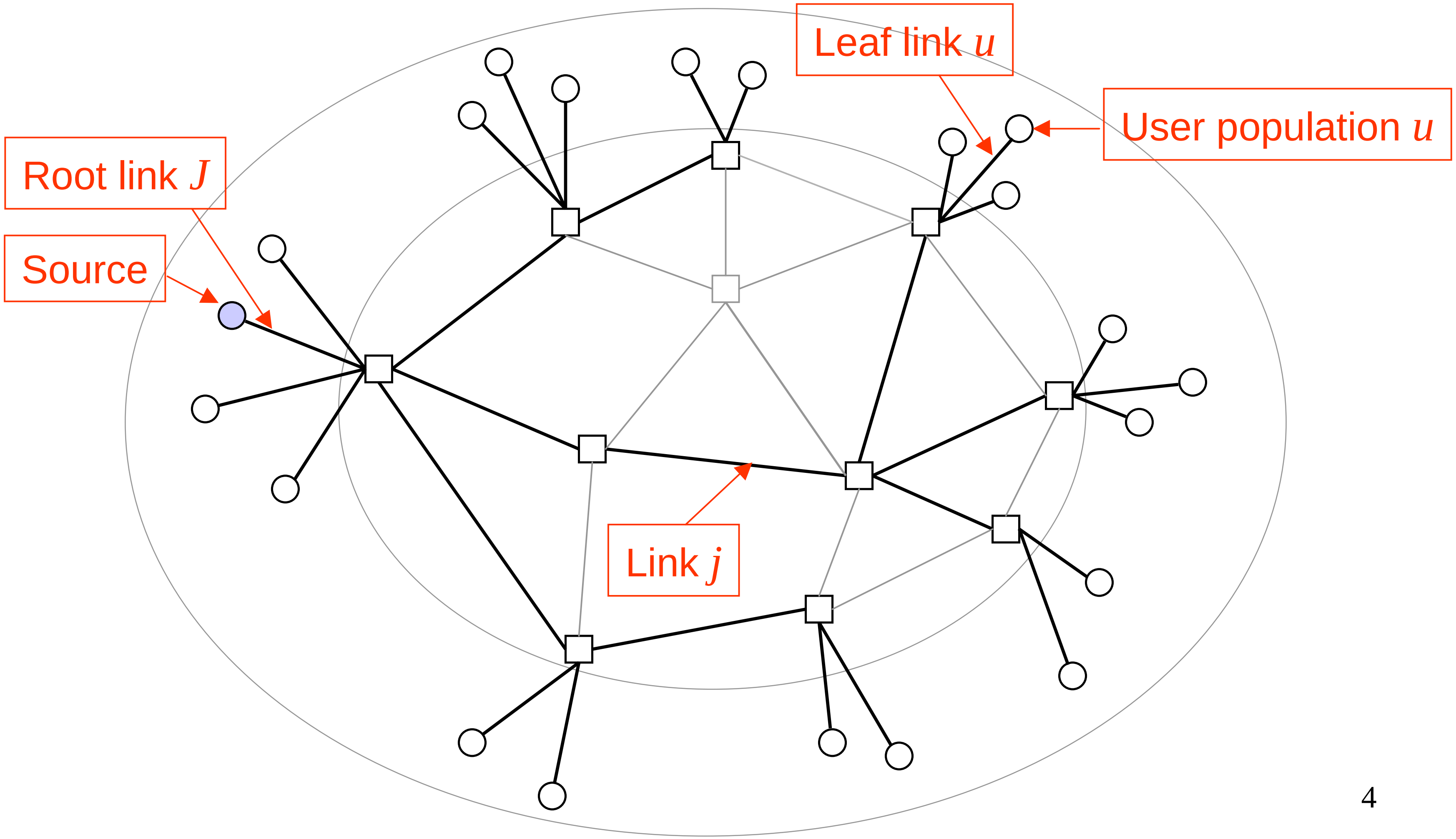
Outline

- Setup: network with hierarchically coded multicast streams
- Three approaches to calculate blocking probabilities
- Combinatorial convolution-truncation algorithm
- Numerical example
- Summary & ongoing work

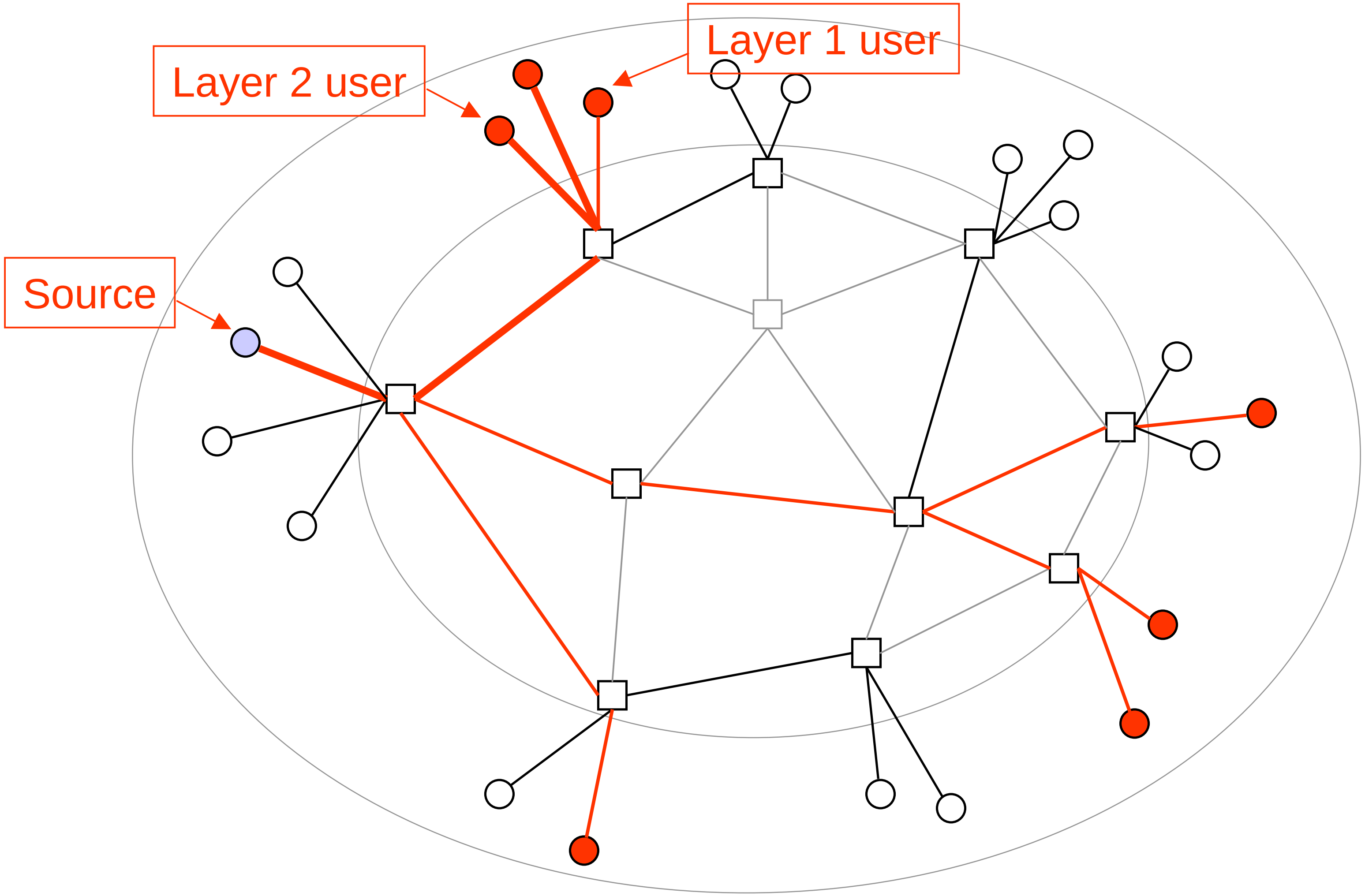
Setup

- Circuit-sw. network, or packet-sw. with strict quality guarantees
- A unique **source** offers a variety of **channels** $i \in I$
  - hierarchically coded audio or video streams with two layers
  - layer 1 = the most important substream, layer 2 = both substreams
  - required capacity $d(l)$ on each link depends on layer l
- Each channel is delivered to user populations $u \in U$ by a **multicast connection** with **dynamic membership**
- Each multicast connection uses the same **routing tree**
  - the source located at the **root node**
  - users located at **leaf nodes**
- Physical links $j \in J$ with finite capacities C_j

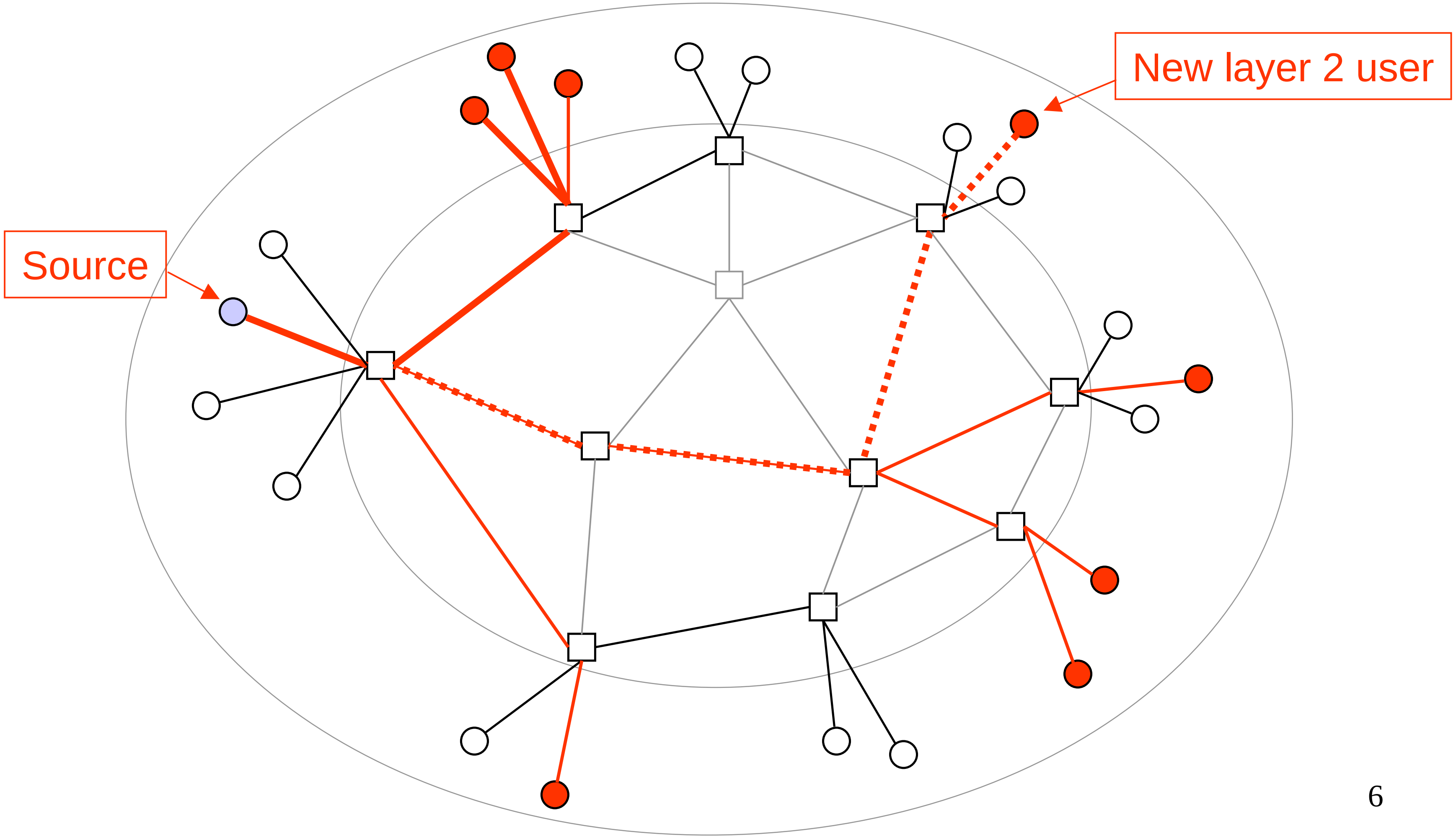
Routing tree



Layered Multicast Connection With Dynamic Membership (1)



Layered Multicast Connection With Dynamic Membership (2)



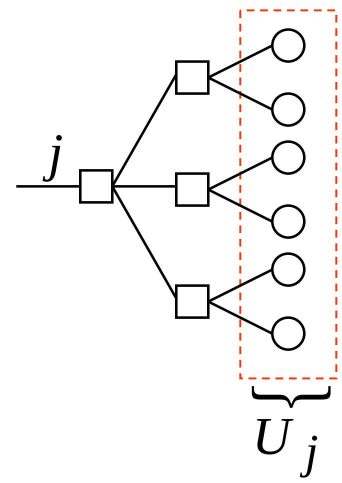
Unlimited link capacities (1)

- Consider first a network with **unlimited** link capacities
- Let

$$Y_{ji} = \text{"state of channel } i \text{ on link } j\text{"} \in \{0,1,2\}$$

- Note that

$$Y_{ji} = \max\{Y_{ui}; u \in U_j\}$$



- Link state (for any link $j \in J$)

$$\mathbf{Y}_j = (Y_{ji}; i \in I) \in S := \{0,1,2\}^I$$

- Network state

$$\mathbf{X} = (\mathbf{Y}_u; u \in U) = (Y_{ui}; u \in U, i \in I) \in \Omega := \{0,1,2\}^{U \times I}$$

Unlimited link capacities (2)

- Assume: **independent** and **infinite** user populations with Poisson request arrivals and exponential holding times
- Then we have

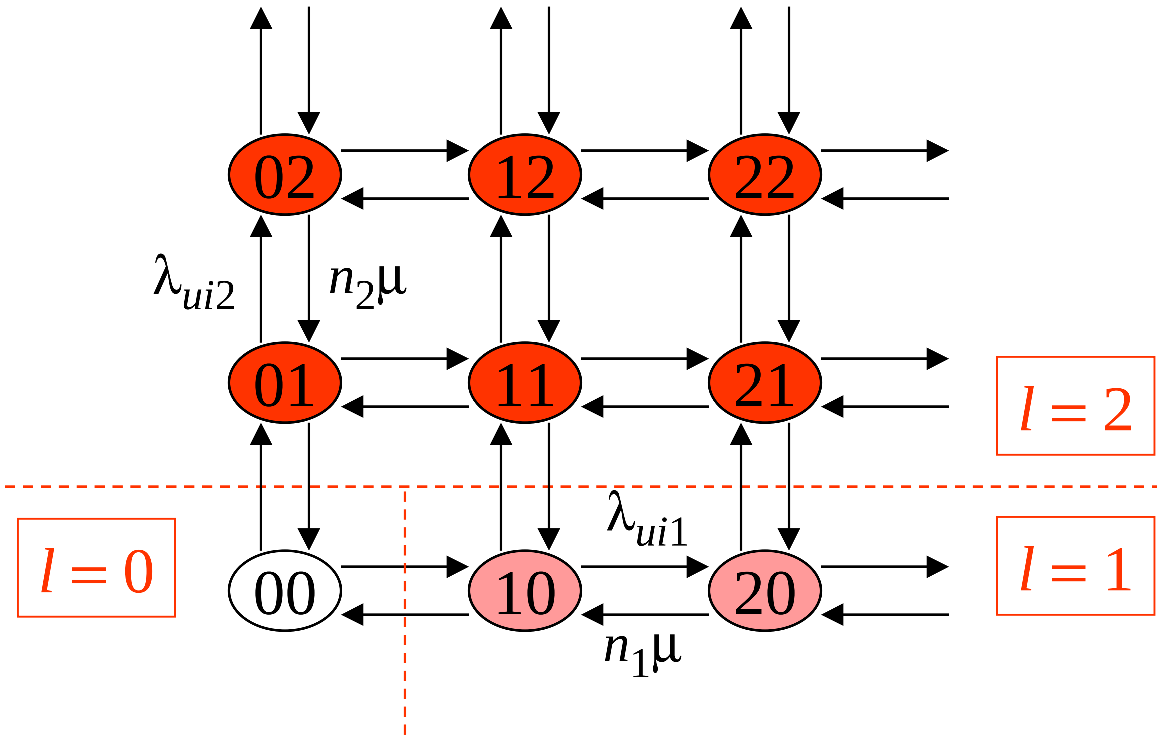
$$P\{\mathbf{X}=\mathbf{x}\}=\prod_{u\in U}P\{\mathbf{Y}_u=\mathbf{y}_u\}=\prod_{u\in U}\prod_{i\in I}P\{Y_{ui}=y_{ui}\}$$

where (by utilising $\mathrm{M}/\mathrm{M}/\infty$ model):

$$p_{uil}:=P\{Y_{ui}=l\}=\begin{cases}\exp(-\frac{\lambda_{ui1}}{\mu})\exp(-\frac{\lambda_{ui2}}{\mu}), & l=0\\ \exp(-\frac{\lambda_{ui2}}{\mu})(1-\exp(-\frac{\lambda_{ui1}}{\mu})), & l=1\\ 1-\exp(-\frac{\lambda_{ui2}}{\mu}), & l=2\end{cases}$$

Unlimited link capacities (3)

- User population model
  (for a single channel at a single leaf node)
  - two independent M/M/∞ queues



Limited link capacities

- Consider now a network with **limited** link capacities C_j
- The set of possible network states, $\tilde{\Omega}$, is clearly a subset of Ω
- Let $\tilde{\mathbf{X}}$ denote the network state in this case, $\tilde{\mathbf{X}} \in \tilde{\Omega}$
- As the most detailed traffic process (telling how many users are active on each leaf node, channel and layer) is a reversible Markov process, the **Truncation Principle** applies and we have

$$P\{\tilde{\mathbf{X}} = \mathbf{x}\} = \frac{P\{\mathbf{X} = \mathbf{x}\}}{P\{\mathbf{X} \in \tilde{\Omega}\}}$$

- So, due to the Truncation Principle, it is enough to analyse the (much easier) system with unlimited link capacities!

Blocking probability

- B_{ur} = **blocking probability** for user population u, requested channel I and layer r

$$B_{ur} := 1 - P\{\tilde{\mathbf{X}} \in \tilde{\Omega}_{ur}\} = 1 - \frac{P\{\mathbf{X} \in \tilde{\Omega}_{ur}\}}{P\{\mathbf{X} \in \tilde{\Omega}\}}$$

  – where $\tilde{\Omega}_{ur}$ is the set of non-blocking states for (u,I,r)

- How to calculate B_{ur}?

Calculation of blocking probabilities (1)

- **1st approach**: closed form expression

$$B_{ur} = 1 - \frac{\sum_{\mathbf{x} \in \tilde{\Omega}_{ur}} P\{\mathbf{X} = \mathbf{x}\}}{\sum_{\mathbf{x} \in \tilde{\Omega}} P\{\mathbf{X} = \mathbf{x}\}}$$

- **Problem**: computationally extremely complex
  - exponential growth both in U and I ($|\Omega| = 3^{UI}$)

Calculation of blocking probabilities (2)

- **2nd approach**: algorithm based on the (original) link state

$$B_{ur} = 1 - \frac{\sum_{\mathbf{y}\in S} Q'_{J,r}(\mathbf{y})}{\sum_{\mathbf{y}\in S} Q_J(\mathbf{y})}$$

where probabilities $Q'_{J,r}(\mathbf{y})$ and $Q_j(\mathbf{y})$ can be calculated **recursively** (from the root link J back to leaf links u)

- **Problem**: still computationally complex
  - linear growth in U but exponential growth in I ($|S| = 3^I$)

Calculation of blocking probabilities (3)

- **3rd approach**: algorithm based on a **reduced** link state

$$B_{ur} = 1 - \frac{\sum\limits_{\mathbf{k}' \in S'} \sum\limits_{l} Q'_{J,r}(\mathbf{k}', l)}{\sum\limits_{\mathbf{k} \in S'} Q_J(\mathbf{k})}$$

where probabilities $Q_j(\mathbf{k})$ and $Q'_{J,r}(\mathbf{k}',l)$ can be calculated **recursively** (from the root link J back to leaf links u)

- **Problem**: computationally reasonable ($|S'| = (I+1)^2$) but …
… restrictive assumptions have to be made!

Chosen approach

Restrictive assumptions

- (i) Users belong to two groups, according to which layer they subscribe to
- (ii) Channels are chosen with equal probabilities, i.e.,

$$\lambda_{uil} = \frac{\lambda_{ul}}{I} \quad \text{for all } i$$

  - Make channels **statistically indistinguishable** at each layer!
  - But user populations and the network topology may still be unsymmetric

Consequences

- Channels statistically indistinguishable at each layer $\Rightarrow$
  - Whenever there are k channels active at any layer l on any leaf link u, each possible index combination $\{i_1,\ldots,i_k\}$ is equally probable

- This and the independence of the user populations $\Rightarrow$
  - Whenever there are k channels active at any layer l on any link j, each possible index combination $\{i_1,\ldots,i_k\}$ is equally probable

- Thus,
  - Just count the total number of active channels at each layer
  - Utilize combinatorics

Reduced link state

- Consider again a network with **unlimited** link capacities
- Let

$$K_{jl} = \text{"number of channels at layer } l \text{ on link } j\text{"}$$

- Reduced link state (for any link $j \in J$)

$$\mathbf{K}_j = (K_{j1}, K_{j2}) \in S' := \{0,1,\ldots,I\}^2$$

- Due to the restrictive assumptions made, we have a **multinomial** distribution,

$$\pi_u(\mathbf{k}) := P\{\mathbf{K}_u = \mathbf{k}\} = \frac{I!}{k_1!\,k_2!\,(I-k_1-k_2)!} p_{u0}^{I-k_1-k_2} p_{u1}^{k_1} p_{u2}^{k_2}$$

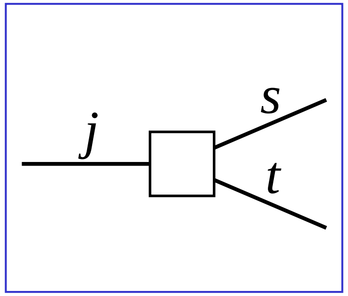
Algorithm (1)

- Key result 1:
  - If link j has two downstream neighbouring links (s,t), then

$$P\{\mathbf{K}_j = \mathbf{k}\} = \sum_{\mathbf{l}} \sum_{\mathbf{m}} \sum_{\mathbf{x}} s(\mathbf{x}, \mathbf{y} \mid \mathbf{l}, \mathbf{m}) P\{\mathbf{K}_s = \mathbf{l}\} P\{\mathbf{K}_t = \mathbf{m}\}$$

  - In other words,

$$\pi_j(\mathbf{k}) = [\pi_s \otimes \pi_t](\mathbf{k})$$



  - Proved by a “sampling without replacement” argument

Algorithm (2)

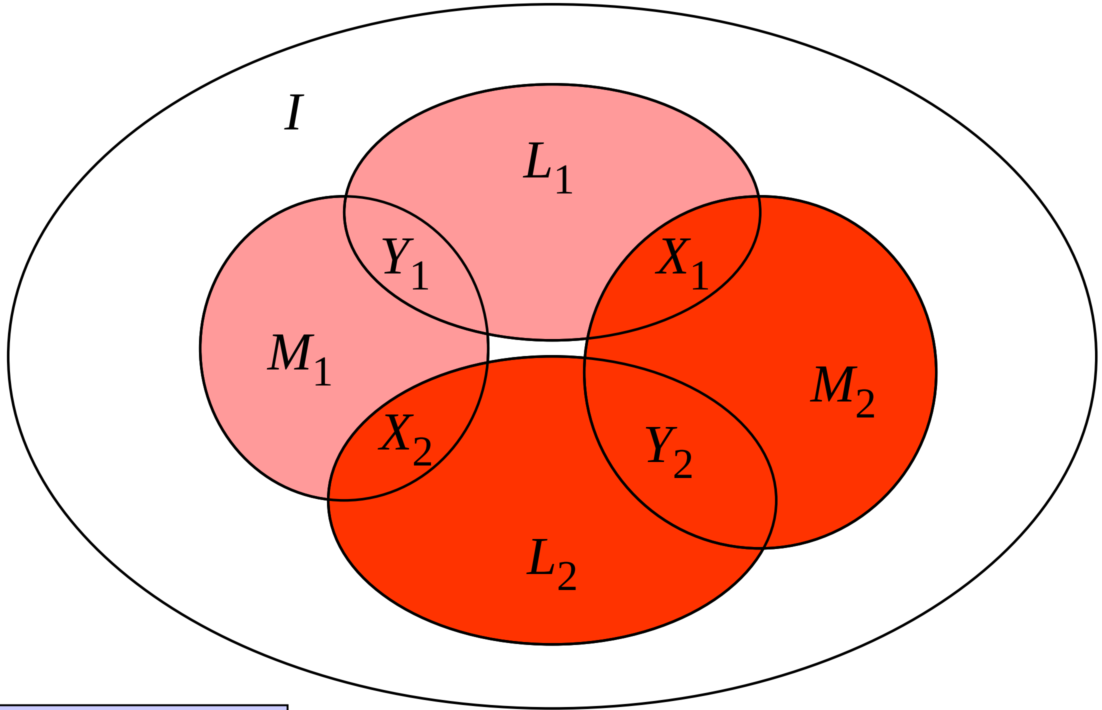
- Definition of **combinatorial convolution** $\otimes$: $R(S')\times R(S') \to R(S')$
  - Let f and g be any real-valued function on $S' = \{0,1,\ldots,I\}^2$.
  - Then define

$$[f \otimes g](\mathbf{k}) = \sum_{\mathbf{l}} \sum_{\mathbf{m}} \sum_{\mathbf{x}} s(\mathbf{x},\mathbf{y} \mid \mathbf{l},\mathbf{m}) f(\mathbf{l}) g(\mathbf{m})$$

where $s(\mathbf{x},\mathbf{y}|\mathbf{l},\mathbf{m})$ is a combinatorial coefficient and vector $\mathbf{y}$ is determined from vectors $\mathbf{k}$, $\mathbf{l}$, $\mathbf{m}$ and $\mathbf{x}$ as follows:

$$\begin{cases} k_1 = l_1 + m_1 - x_1 - x_2 - y_1 \\ k_2 = l_2 + m_2 - y_2 \end{cases}$$

Algorithm (3)



$K_1 = (L_1 \setminus X_1) \cup (M_1 \setminus X_2)$
$K_2 = L_2 \cup M_2$

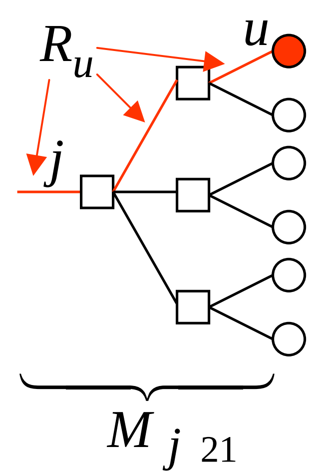
Algorithm (4)

- Define (for all $j \in J$)

$$Q_j(\mathbf{k}) = P\{\mathbf{K}_j = \mathbf{k}; D_{j'} \le C_{j'},\ j' \in M_j\}$$

$$Q'_{jr}(\mathbf{k}', l) = P\{\mathbf{K}'_j = \mathbf{k}', Y_{j,I} = l; D'_{j'} + d(r) \le C_{j'},\ j' \in M_j \cap R_u; D_{j'} \le C_{j'},\ j' \in M_j\}$$

  – where $\mathbf{K}'_j$ is the reduced link state without channel I, $D_{j'}$ is the total demand and $D'_{j'}$ is the demand without channel I



- Then the blocking probability for class (u,I,r) is

$$B_{ur} = 1 - \frac{P\{\mathbf{X} \in \tilde{\Omega}_{ur}\}}{P\{\mathbf{X} \in \tilde{\Omega}\}} = 1 - \frac{\sum_{\mathbf{k}' \in S'} \sum_{l} Q'_{J,r}(\mathbf{k}', l)}{\sum_{\mathbf{k} \in S'} Q_J(\mathbf{k})}$$

Algorithm (5)

- **Recursion 1** to calculate the denominator $Q_J(\mathbf{k})$:

$$Q_j(\mathbf{k}) = \begin{cases} T_j[\pi_j](\mathbf{k}), & j \in U \\ T_j[\bigotimes_{j' \in N_j} Q_{j'}](\mathbf{k}), & j \notin U \end{cases}$$

- Definition of **truncation** operator T_j: $R(S') \to R(S')$
  - Let f be any real-valued function on $S' = \{0,1,\ldots,I\}^2$.
  - Then define

$$T_j[f](\mathbf{k}) = f(\mathbf{k}) \cdot 1\{k_1 d(1) + k_2 d(2) \le C_j\}$$

Algorithm (6)

- **Recursion 2** to calculate the numerator $Q'_{J,r}(\mathbf{k}',l)$:

$$Q'_{jr}(\mathbf{k}',l) = \begin{cases} T^{\circ}_{ur}[E\pi_u](\mathbf{k}',l), & j = u \\ T^{\circ}_{jr}[Q'_{D_u(j),r} \odot \bigotimes_{j' \in N_j \setminus R_u} Q_{j'}](\mathbf{k}',l), & j \in R_u \setminus \{u\} \end{cases}$$

- Definition of **truncation** operator T°_{jr}: $R(S'') \to R(S'')$
  - Let f be any real-valued function on $S'' = S' \times \{0,1,2\}$.
  - Then define

$$T^{\circ}_{jr}[f](\mathbf{k}',l) = f(\mathbf{k}',l) \cdot 1\{d(1)k'_1 + d(2)k'_2 + \max\{d(l), d(r)\} \le C_j\}$$

Algorithm (7)

- Definition of operation $\odot$: $R(S'') \times R(S') \to R(S'')$
  - Let f and g be any real-valued function on S'' and S', respectively
  - Then define

$$[f \odot g](\mathbf{k}', l) = \sum_{\mathbf{l}'} \sum_{\mathbf{m}'} \sum_{\mathbf{x}} s'(\mathbf{x}, \mathbf{y} \mid \mathbf{l}', \mathbf{m}') \sum_{\max\{v_1, v_2\} = l} f(\mathbf{l}', v_1) E g(\mathbf{m}', v_2)$$

where $s'(\mathbf{x},\mathbf{y}|\mathbf{l}',\mathbf{m}')$ is another combinatorial coefficient and vector $\mathbf{y}$ is determined from vectors $\mathbf{k}'$, $\mathbf{l}'$, $\mathbf{m}'$ and $\mathbf{x}$ as before

Algorithm (8)

- Key result 2:
  - If link j has two downstream neighbouring links (s,t), then

$$P\{\mathbf{K}'_j = \mathbf{k}\} = \sum_{\mathbf{l}'} \sum_{\mathbf{m}'} \sum_{\mathbf{x}} s'(\mathbf{x}, \mathbf{y} \mid \mathbf{l}', \mathbf{m}') P\{\mathbf{K}'_s = \mathbf{l}'\} P\{\mathbf{K}'_t = \mathbf{m}'\}$$

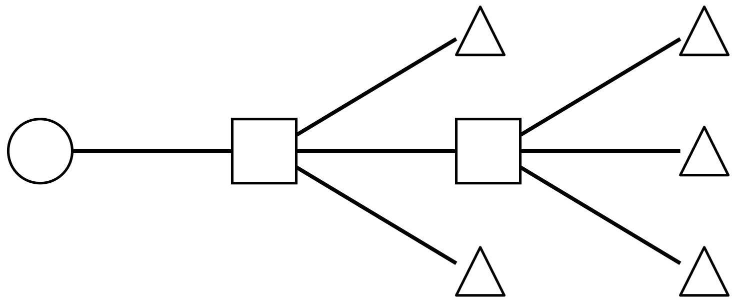
  - So, $s'(\mathbf{x},\mathbf{y}|\mathbf{l}',\mathbf{m}')$ is the same as $s(\mathbf{x},\mathbf{y}|\mathbf{l},\mathbf{m})$ but without channel I
- Definition of operator $E : R(S') \rightarrow R(S'')$
  - Let f be any real-valued function on S'.
  - Then define

$$E[f](\mathbf{k}', l) = \begin{cases} \dfrac{I - k'_1 - k'_2}{I} f(\mathbf{k}'), & l = 0 \\ \dfrac{k'_l + 1}{I} f(\mathbf{k}' + \mathbf{e}_l), & l > 0 \end{cases}$$

Example network

- Comparison of execution times:

I	time [s]
4	4
8	32
12	228
16	1191
20	4727
24	15386



Summary and ongoing work

- Summary:
  - “Combinatorial convolution-truncation” algorithm presented for the calculation of blocking probabilities in a network with hierarchically coded multicast streams
    - approximate complexity $O(UI^8)$
    - avoids exponential dependence on U and I but …
    - … requires restrictive assumptions (all channels have to look the same)
- Ongoing work:
  - generalisation of the algorithm for more layers, more groups of multicast channels, and other user population models (incl. general holding time distribution)
  - development of an efficient simulation method (by the Inverse Convolution approach)

The End