Dynamic resource allocation for virtual path connections

Samuli Aalto
Laboratory of Telecommunications Technology
Helsinki University of Technology

samuli.aalto@hut.fi

Background

- Work done at VTT in WP3.2 of the ACTS project EXPERT (from January 96 until January 97)
  - “Initial specification of bandwidth management”, Deliverable 4, March 96
  - “Interim report on bandwidth and routing trials”, Deliverable 8, January 97
  - “Virtual Trunk simulation”, EXPERT ATM Traffic Symposium, September 97, together with S. Giordano (EPFL)
- Work done at VTT and HUT in COST 257 project
  - “Blocking probabilities in a transient system”, COST257 TD(97)14, January 97, together with J. Virtamo
  - “Remarks on the effectiveness of dynamic VP bandwidth management”, COST257 TD(97)15, January 97, together with J. Virtamo

Contents

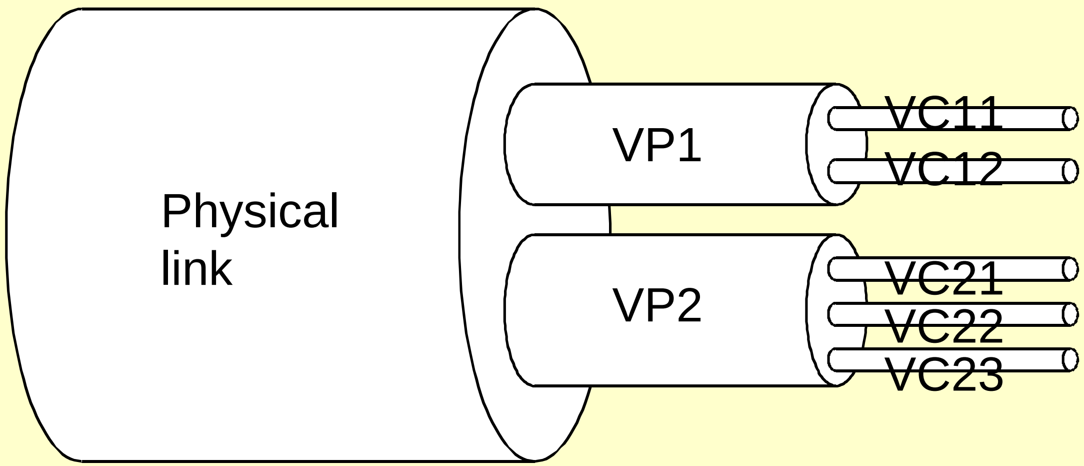
- Introduction to virtual paths and their management
- Periodic capacity allocation scheme
- CAC functions
- Simulation results
- Summary and future work

Introduction

- ATM offers two types of connections:
  - Virtual Channel Connections (VCC)
  - Virtual Path Connections (VPC)
- Virtual Channel Connections
  - established on demand by users (signalling)
- Virtual Path Connections
  - established semi-permanently by network (network management)

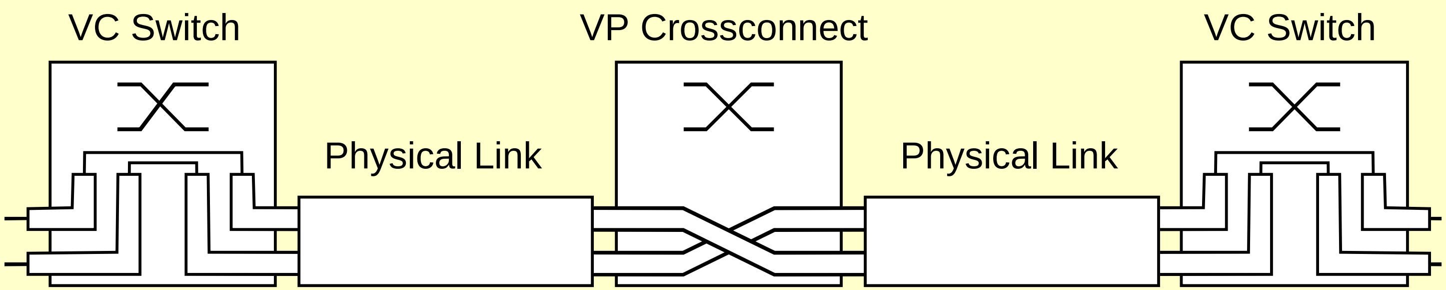
Virtual Paths

- each Virtual Path carries a bunch of Virtual Channels



Why Virtual Paths?

- to reduce the processing needed to establish individual VCCs
- to separate traffic classes with different QoS/GoS requirements
- to increase flexibility by creating logical networks
- to help in managing networks



Why not Virtual Paths?

- VPCs usually static and of the CBR type
  - ⇒ statistical multiplexing possible only within VP, not between separate VPs
  - ⇒ statistical multiplexing gain reduced

VPC management (slower time scale)

- Network design problem
  - given the traffic demands per each class and OD-pair
    - which nodes should be connected with VPCs?
    - how should the VPCs be routed?
    - what should their (initial) capacities be?

  ⇒ construction of all VPCs
- Solutions for the routing and dimensioning problem
  - Generalization of Gopal's method by Arvidsson (1994)
  - Stochastic Allocation by Pioro and Gajowniczek (1994)

VPC management (faster time scale)

- Dynamic capacity allocation problem
  - given the VPCs, their routes and states (i.e. the number of established VCCs per each VPC)
    - what should their (updated) capacities be?

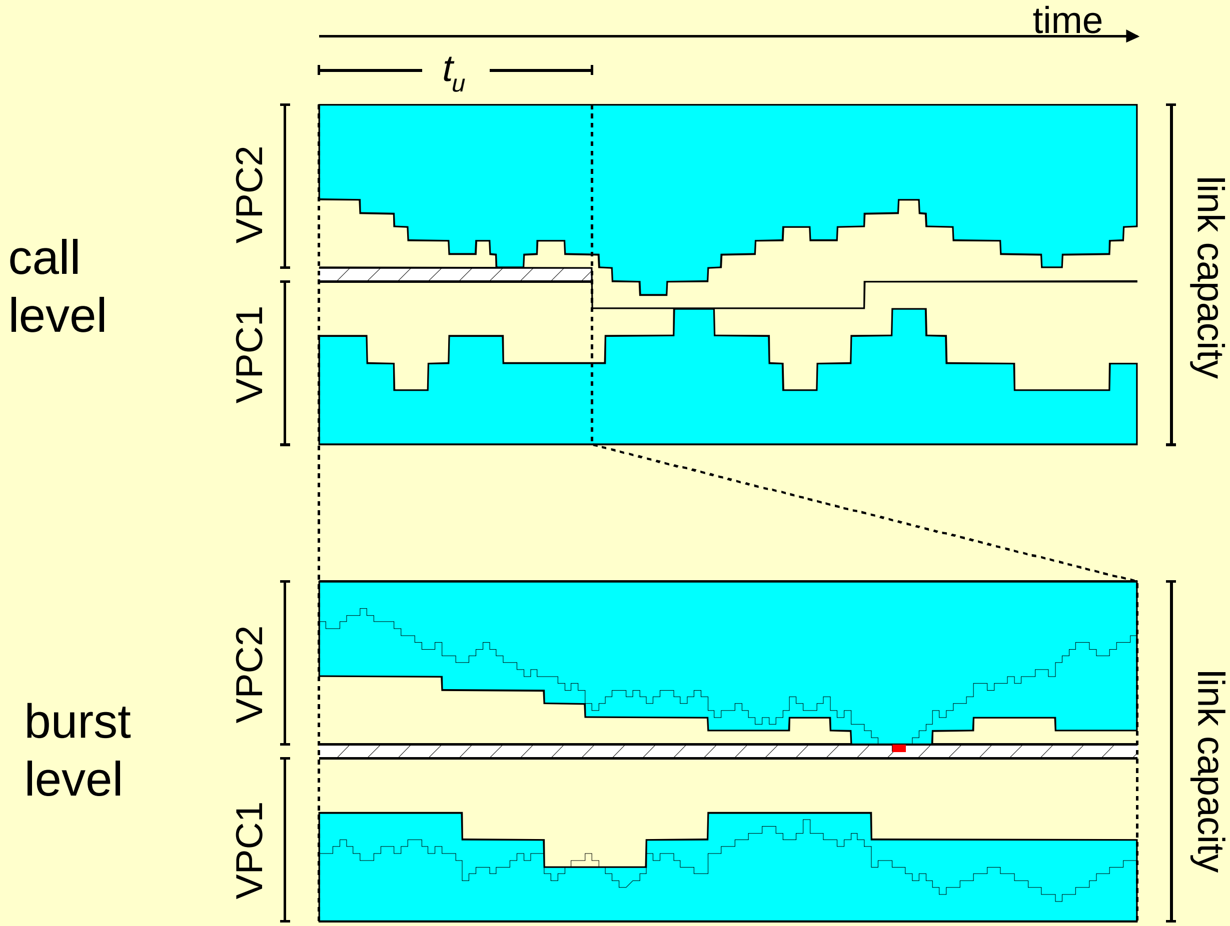
  $\Rightarrow$ reallocation of VPC capacities
- Solutions for the dynamic capacity allocation problem
  - State-dependent capacity allocation scheme (with thresholds) by Ohta and Sato (1992)
  - Periodic capacity allocation scheme by Mocci et al. (1994-)

Contents

Periodic capacity allocation scheme

- Originally proposed by Mocci et al. (1994-)
- Based on the knowledge of the number of active VCC connections
- Parameter
  - updating interval t_u of VPC capacities
- Assumptions
  - traffic classified into a finite set of traffic classes
  - each VPC devoted to a single traffic class (traffic separation approach)
  - traffic classes specified by
    - mean holding time T ($\Delta = T/t_u$) for a connection (exponential)
    - bandwidth demand derived from the traffic descriptor and QoS requirement (cf. CAC functions)
    - GoS requirement ε (constraining call blocking)
  - Poisson arrivals of connection requests

Processes in different time scales



Idea

- first VPC-by-VPC in all links separately
  - allocate as much capacity as is needed to satisfy the class-specific call blocking target during the next updating interval
  - requires the calculation of the transient blocking probability during the next updating interval given the initial number of established VCC connections
- then link-by-link
  - if the total capacity of a link is exceeded, adjust the VPC capacities in a “fair” way
  - requires the definition of fair share
- finally network-wide (optional)
  - allocate to VPCs the minimum capacity along the physical path

Details (1)

- Consider a VPC with offered load ρ passing through link l and carrying n VCC connections belonging to class k. Calculate first

$$N = \min\{ m \mid \text{transientErlang}(m, \rho, n, \Delta_k) \le \varepsilon_k \}$$

  – Simple approximation used in the simulations:

$$N = n \exp(-\Delta_k) + N^{\text{stat}}(\rho, \varepsilon_k)(1 - \exp(-\Delta_k))$$

$$N^{\text{stat}}(\rho, \varepsilon_k) = \min\{ m \mid \text{Erlang}(m, \rho) \le \varepsilon_k \}$$

  – In fact, it would be nice to have a **robust** allocation function depending on n, Δ, ε but not on ρ.

- Anyway, transform N to the capacity requirement C by the suitable CAC function:

$$C = \text{requiredCapacity}_k(N)$$

Details (2)

- Sum up the capacities C_k of all VPCs passing through link l (indexed by their classes k), and compare the result to the link capacity C^l.
- Allocate these capacities to the VPCs, provided that

$$\sum_k C_k \leq C^l$$

- Otherwise, adjust the capacities as follows.
- Calculate first for each VPC the capacity required by the existing VCCs:

$$c_k = \text{requiredCapacity}_k(n_k)$$

- Calculate then the remaining capacity R:

$$R = C^l - \sum_k c_k$$

Details (3)

- Define the fair share to be:

$$f_k = \frac{C_k - c_k}{\sum_{k'} C_{k'} - c_{k'}}$$

- Adjusted capacities are

$$\tilde{C}_k = c_k + R f_k$$

- However, consumed capacities will be

$$\hat{C}_k = \mathrm{requiredCapacity}_k(\mathrm{allowedNrCalls}_k(\tilde{C}_k))$$

- Thus, there is still some bandwidth to be shared (iterate!):

$$\tilde{R} = C^l - \sum_k \hat{C}_k$$

Bounds for efficiency (1)

- With fixed offered load ρ, the average allocation of the periodic allocation scheme lies between that of
  - the static allocation scheme and
  - the fully dynamic allocation scheme
- Static capacity allocation
  - made once for all (realizing Complete Partitioning policy)
  - constant capacity for $N^{stat}(\rho,\varepsilon)$ VCCs reserved for each VPC such that the average blocking probability is below the target level ε
  - average allocation in the ideal case = $N^{stat}(\rho,\varepsilon)$
- Fully dynamic (on-demand) capacity allocation
  - made every time a new VCC is established or torn-down (realizing Complete Sharing policy)
  - better performance but also much more overhead
  - average allocation in the ideal case = ρ

Bounds for efficiency (2)

- Example ($\varepsilon = 0.01$): with offered load $\rho = 10$ (erlang) upper bound for the attainable saving is 45 % but with $\rho = 1000$ (erlang) only 3 %

ρ	$N^{stat}(\rho,\varepsilon)$
10	18
100	117
1000	1029

- Note: Ideal case means infinite (physical) link capacity. Thus in practice, with finite link capacity, the saving is still smaller.
- On the other hand, the static allocation should be based on “busy hour” load, whereas dynamic allocation utilizes “current” load (which even needs not to be known when robust allocation methods used) .

Contents

- Introduction to virtual paths and their management
- Periodic capacity allocation scheme
- CAC functions
- Simulation results
- Summary and future work

CAC functions

- given the number of connections *N* belonging to class *k*,
  - calculate the required bandwidth *C*

$$C = \mathrm{requiredCapacity}_k(N)$$

- given the bandwidth *C* devoted to class *k*,
  - calculate the allowed number of connections *N*

$$N = \mathrm{allowedNrCalls}_k(C)$$

- given the former, the latter is

$$\mathrm{allowedNrCalls}_k(C) = \max\{m \mid \mathrm{requiredCapacity}_k(m) \le C\}$$

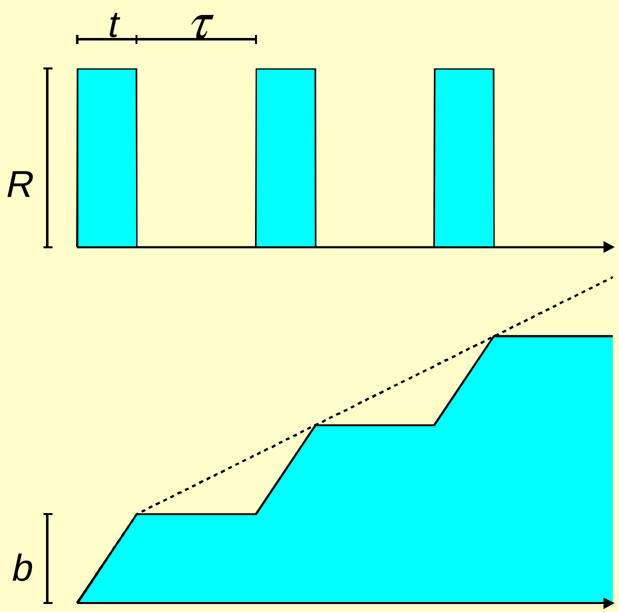
Assumptions

- traffic comes from VBR sources
- VBR traffic classes described by three parameters (burst scale fluid approximation)
  - peak rate R
  - sustainable rate m
  - burst tolerance τ in time units
    - max burst length $t = \tau m/(R - m)$ in time units
    - burst tolerance $b = \tau m = t(R - m)$ in information units

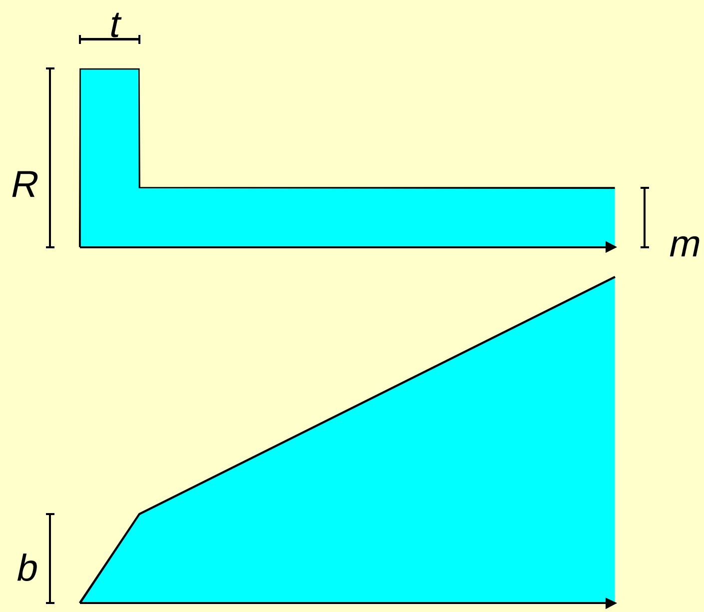
  $\Rightarrow$ traffic corforms to
    - GCRA($1/R$,0) and GCRA($1/m,\tau$)
- QoS: **lossless** transfer even in the (deterministic) worst case with synchronized sources
- Each VPC has a shaping buffer of size B

Worst case (deterministic) VBR traffic source

- Worst case periodic source



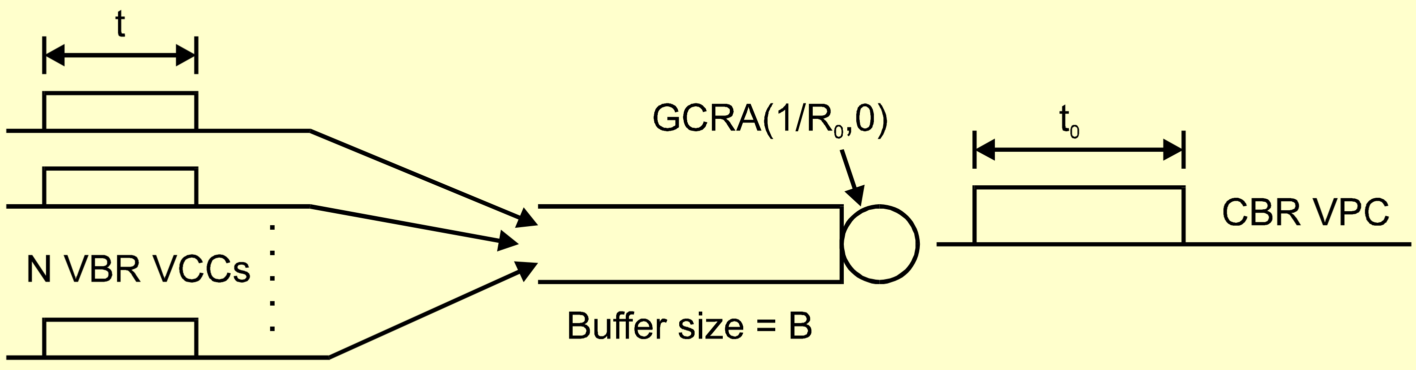
- Ultimately worst case source



Two approaches

- **VBR-over-CBR**
  - VPCs of the CBR type
  - CBR VPC characterized by one parameter
    - peak rate R_0
- **VBR-over-VBR** (proposed by Giordano et al. (1997))
  - VPCs of the VBR type
  - VBR VPC characterized by three parameters
    - peak rate R_0
    - sustainable rate m_0
    - burst tolerance τ_0 in time units
      (or max burst length $t_0 = \tau_0 m_0/(R_0 - m_0)$ in time units)

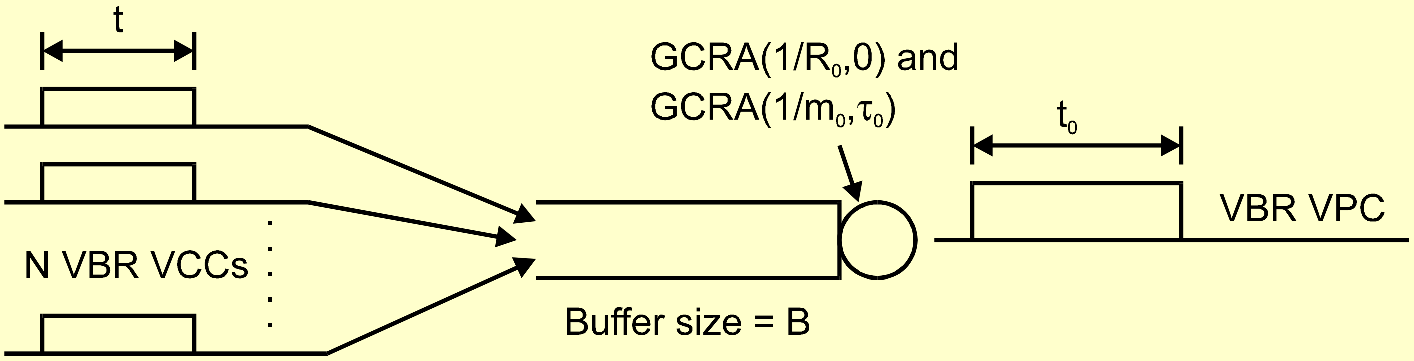
VBR-over-CBR



- requiredCapacity(N) = $R_0(N)$

$$R_0(N) = \max\{NR - B/t, Nm\}$$

VBR-over-VBR



- requiredCapacity(N) = equivalentCapacity($R_0(N), m_0(N), \tau_0(N), B', \delta$) = min!

$$R_0(N) = \max\{NR - B/t, Nm\}$$

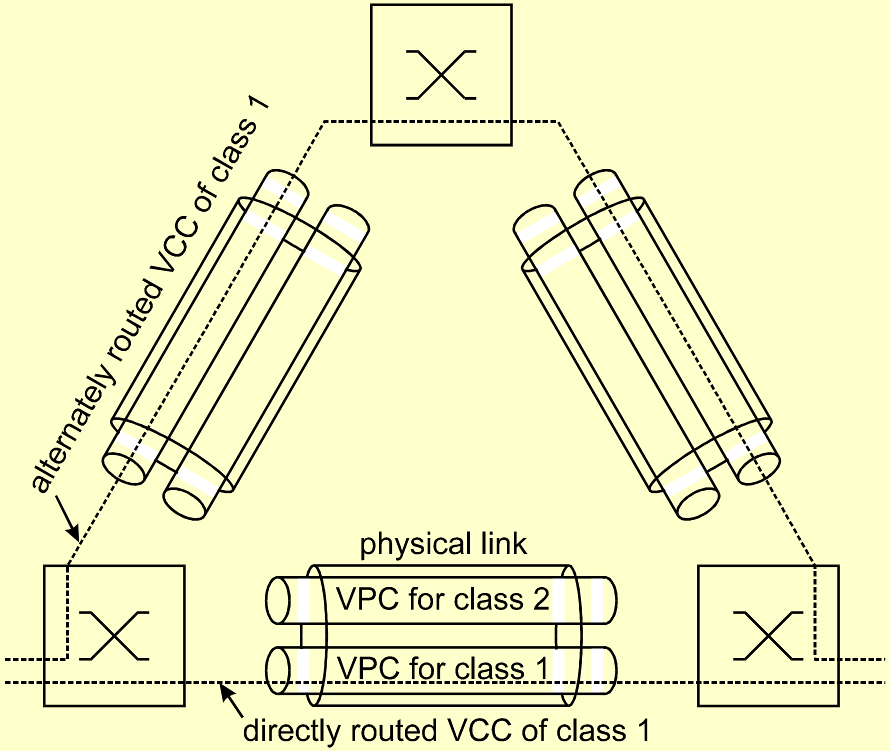
$$m_0(N) = Nm$$

$$t_0(N) = NRt / R_0(N)$$

- equivalentCapacity(...) approximated as in Guerin et al. (1991)
- more precisely (lossless!) by applying Elwalid et al. (1995)

Contents

Simulation scenario



Two traffic classes: broadband and narrowband

Param.	bb class	nb class
R	10.0	1.0
m	2.0	0.2
τ	80	80
t	20	20
ε	0.01	0.01
δ	0.00001	0.00001
T	1	1

Simulation trials

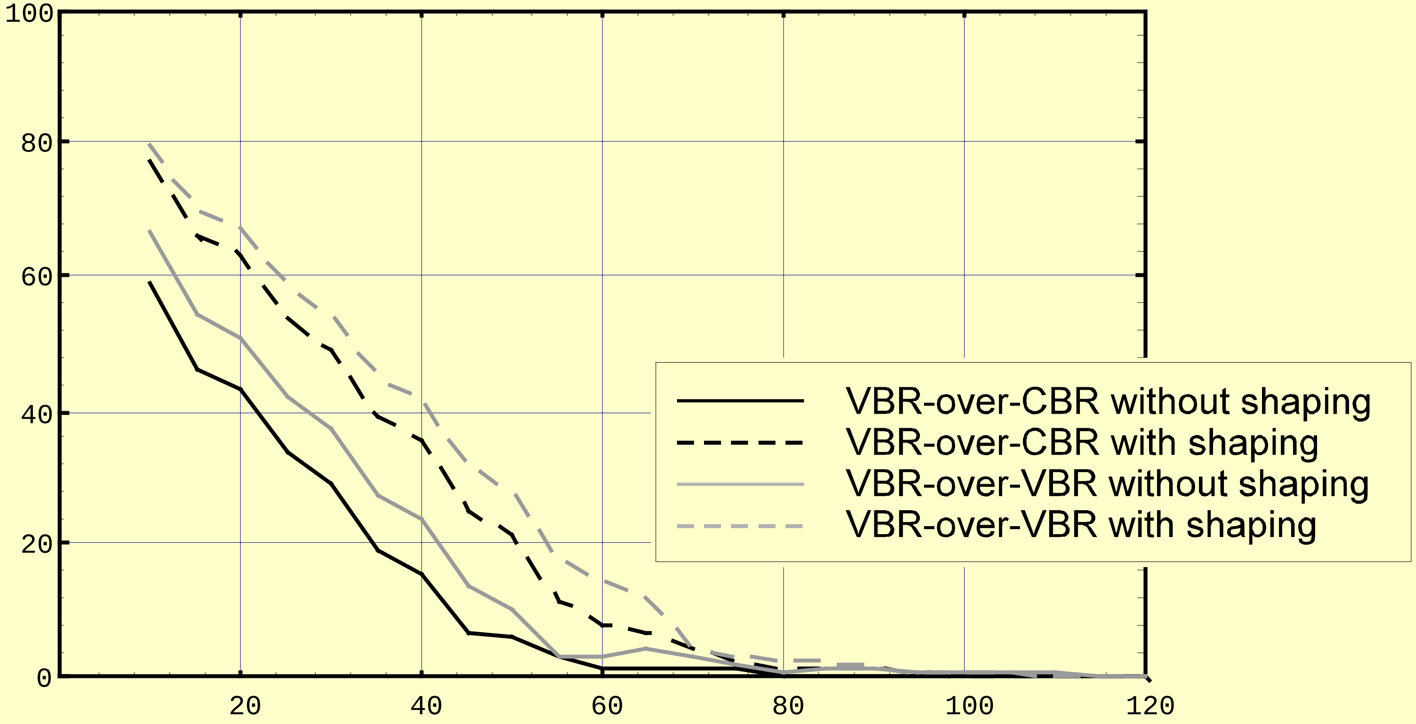
- Trial 1: VBR-over-CBR vs. VBR-over-VBR
  - shaping buffer B = 0/200 cells, link buffer B^l = 0/1000 cells
  - direct routing
  - $t_u = 1.0$
- Trial 2: Direct vs. alternative routing
  - VBR-over-CBR with no shaping
  - direct/alternative routing
  - $t_u = 1.0$
- Trial 3: Varying updating interval
  - VBR-over-CBR with no shaping
  - direct routing
  - $t_u = 0.1/0.5/1.0$

Simulation conclusions

- Trial 1: VBR-over-CBR vs. VBR-over-VBR
  - VBR-over-VBR results in better performance (as expected)
  - the difference does not seem to be very large
    - partly due to the rather inefficient method for calculating the "effective bandwidth"
  - even better results by introducing shaping buffers
    - requires that the traffic shaped is not critical for delays
- Trial 2: Direct vs. alternative routing
  - with light and medium traffic, alternative routing performs better
  - with heavy traffic, direct routing is slightly better
    - larger networks make a clearer distinction
- Trial 3: Varying updating interval
  - the simple (approximative) allocation function works (only) if the updating interval is great enough (1 holding time)

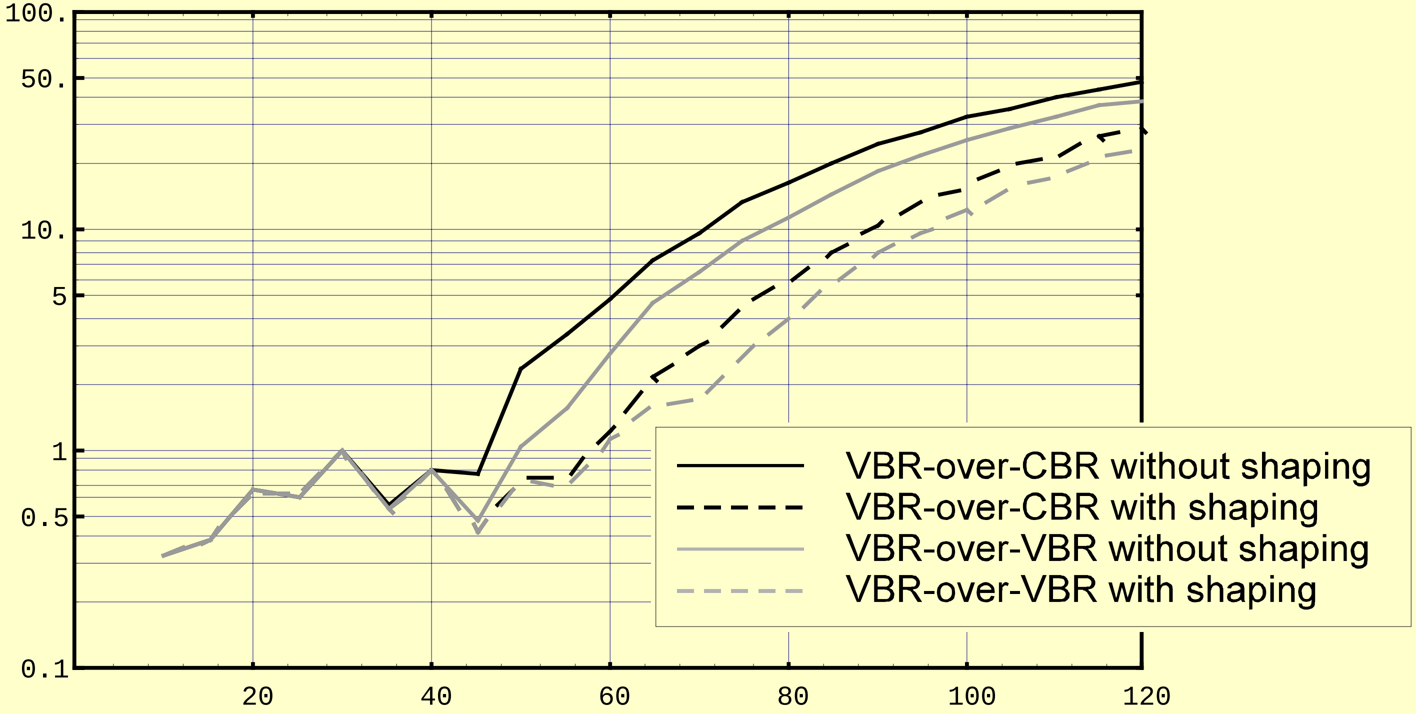
Trial 1: VBR-over-CBR vs. VBR-over-VBR (1)

- Percentage of average free capacity vs. traffic load



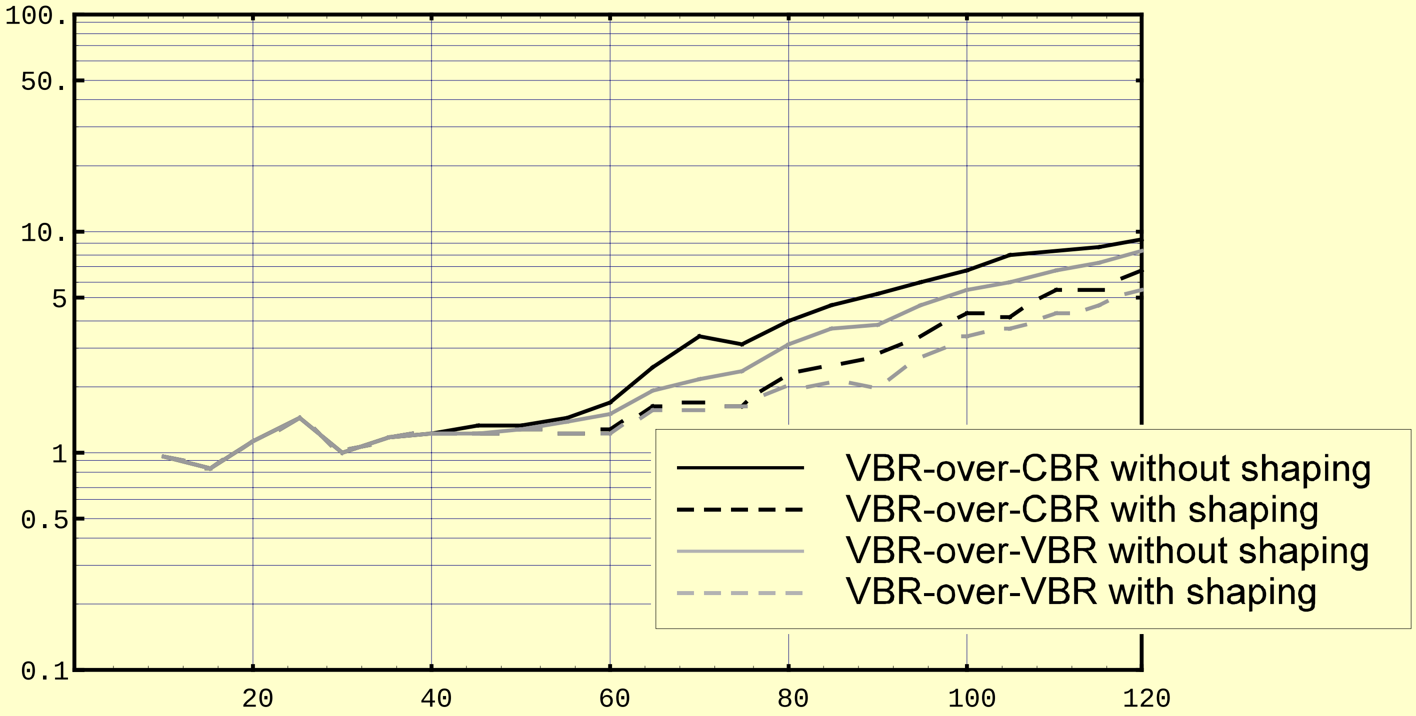
Trial 1: VBR-over-CBR vs. VBR-over-VBR (2)

- Percentage of rejected calls (bb class) vs. traffic load



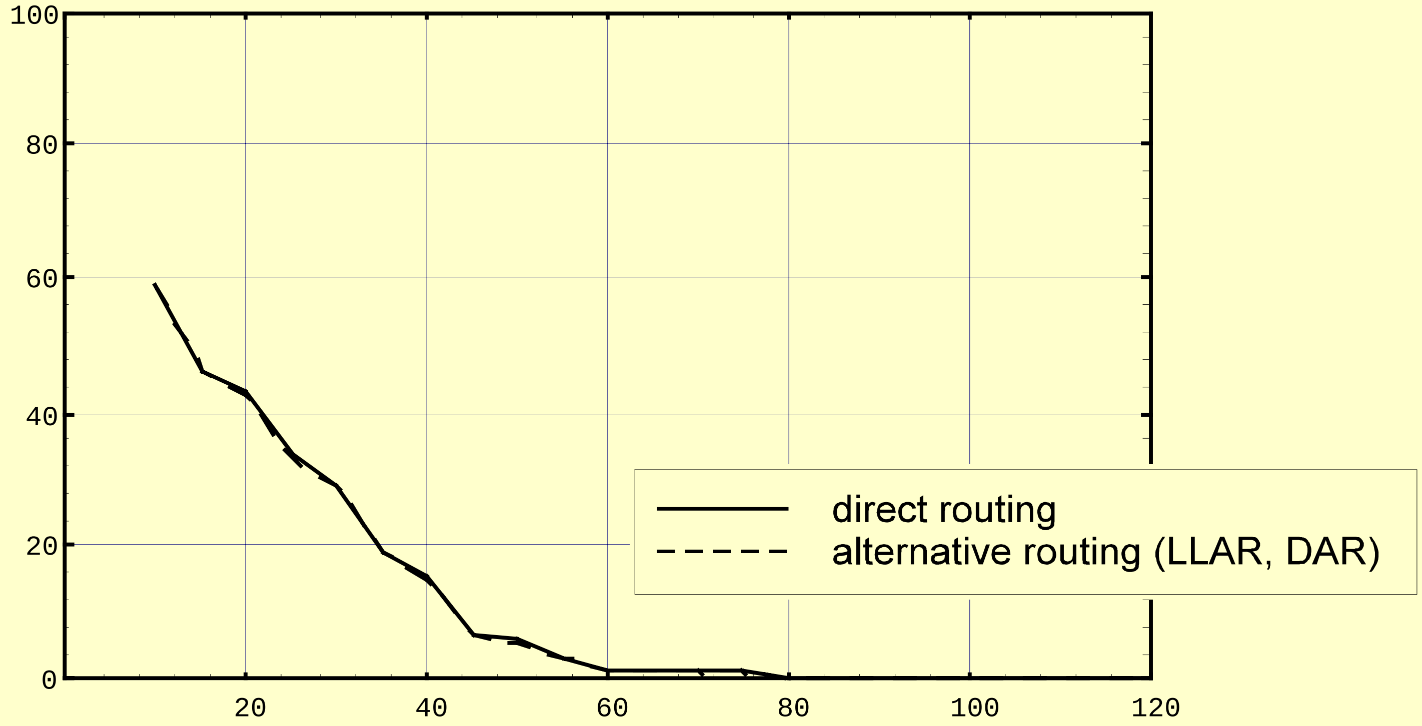
Trial 1: VBR-over-CBR vs. VBR-over-VBR (3)

- Percentage of rejected calls (nb class) vs. traffic load



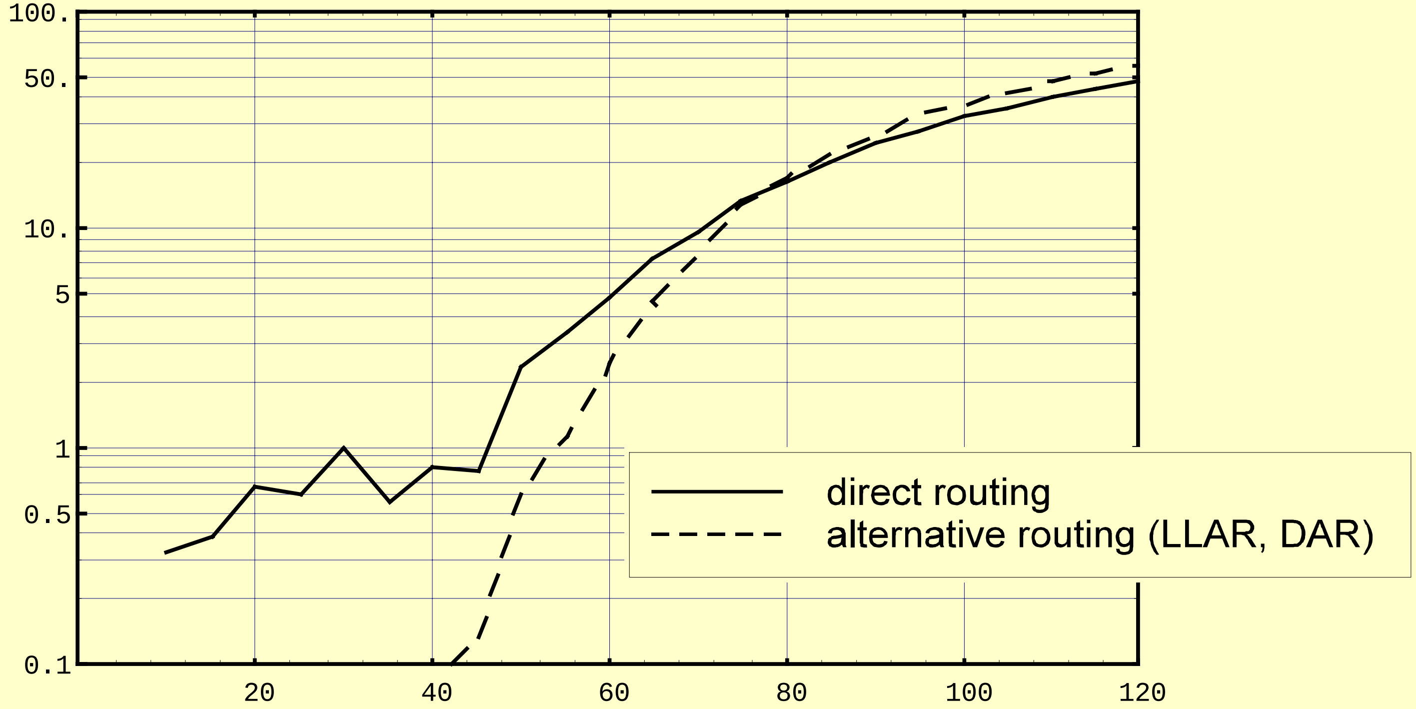
Trial 2: Direct vs. alternative routing (1)

- Percentage of average free capacity vs. traffic load



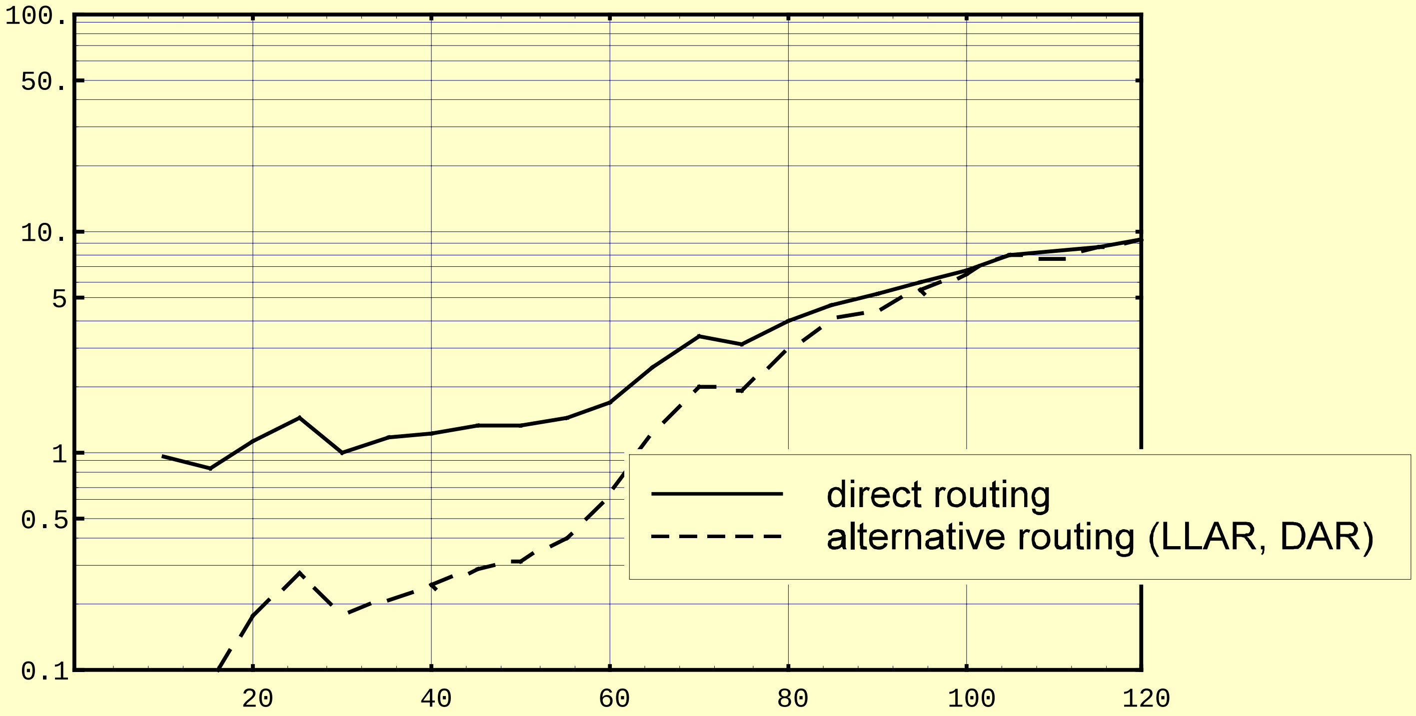
Trial 2: Direct vs. alternative routing (2)

- Percentage of rejected calls (bb class) vs. traffic load



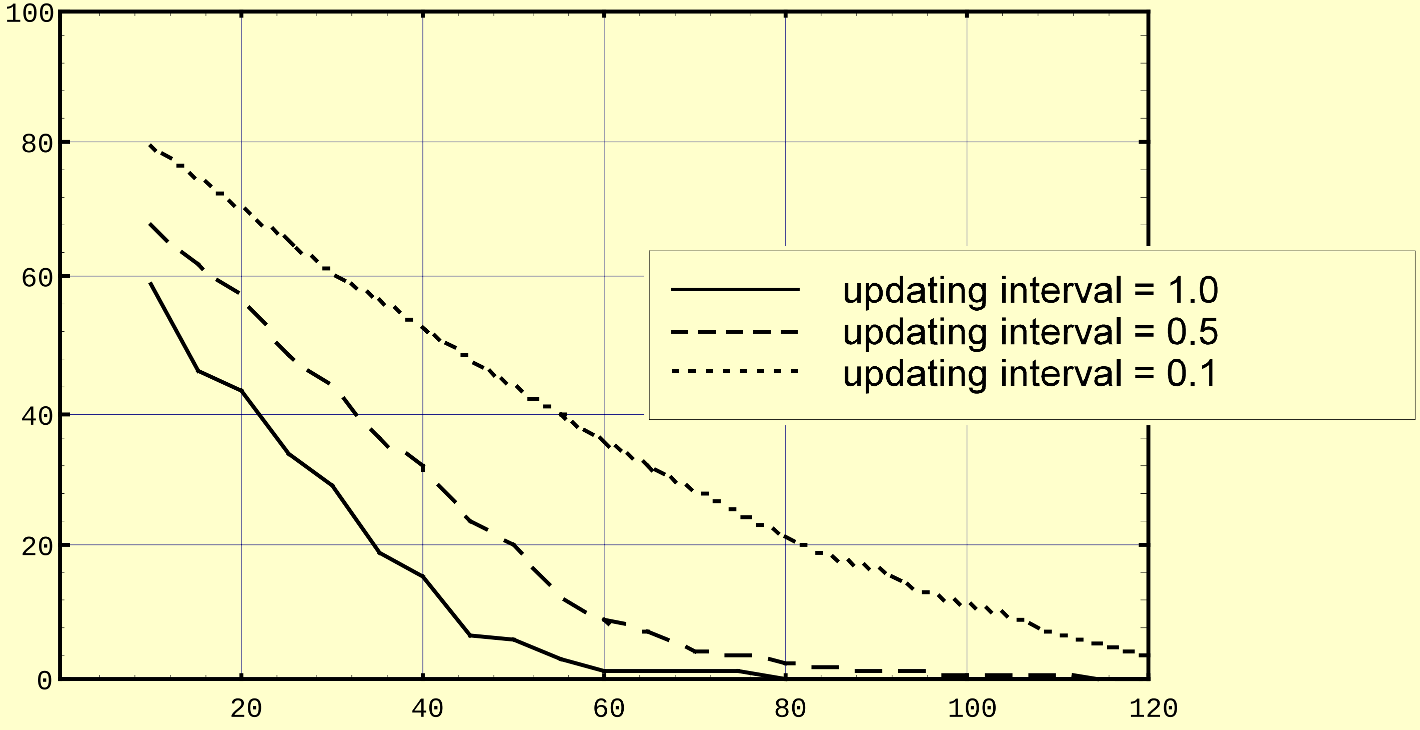
Trial 2: Direct vs. alternative routing (3)

- Percentage of rejected calls (nb class) vs. traffic load



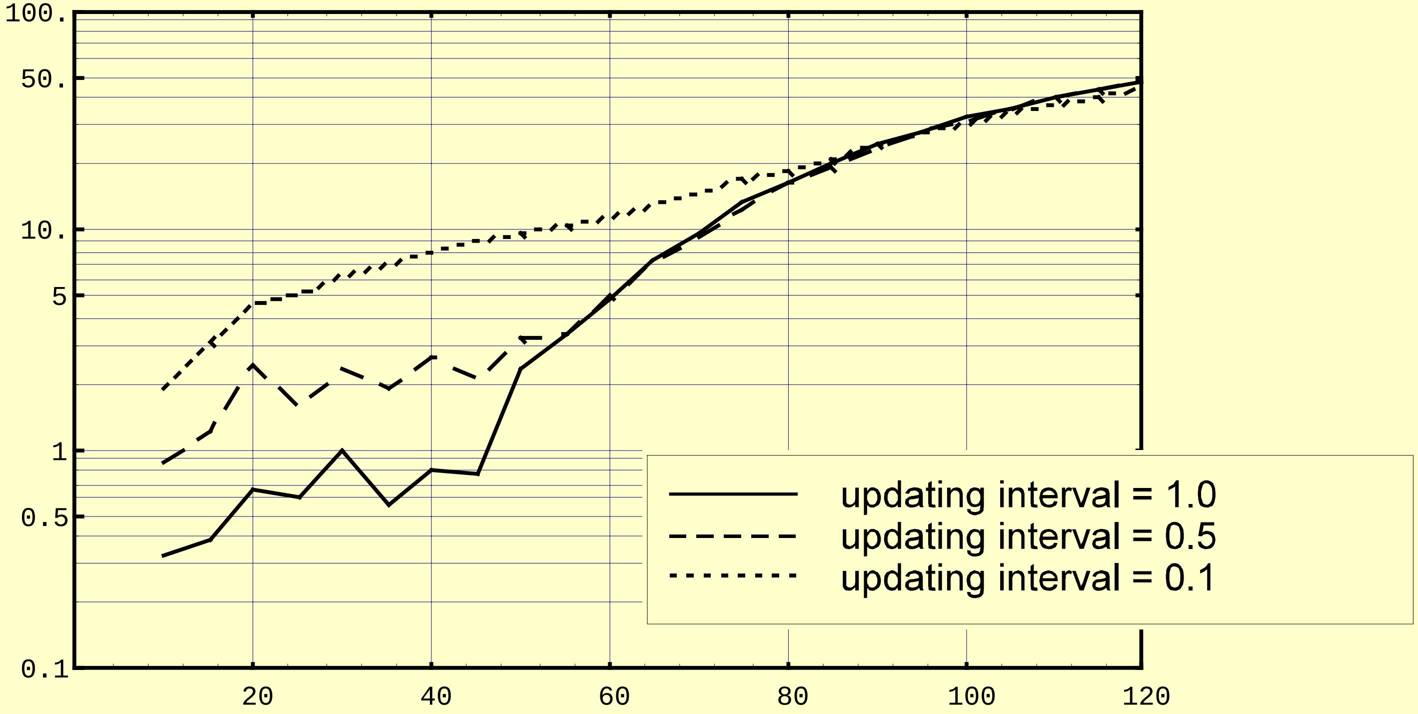
Trial 3: Varying updating interval (1)

- Percentage of average free capacity vs. traffic load



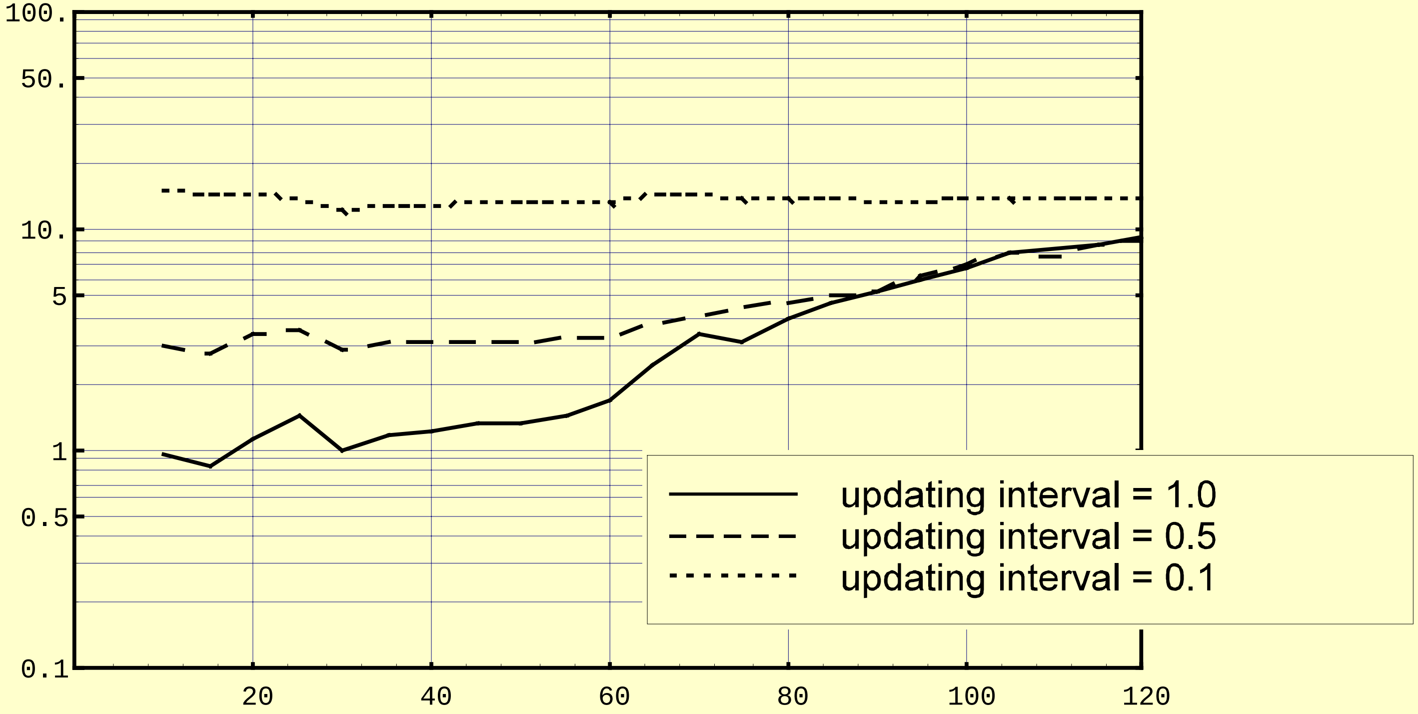
Trial 3: Varying updating interval (2)

- Percentage of rejected calls (bb class) vs. traffic load



Trial 3: Varying updating interval (3)

- Percentage of rejected calls (nb class) vs. traffic load



Contents

Summary and future work

- Summary
  - dynamic capacity allocation scheme for VPCs was reviewed
  - simple allocation function was presented
  - allocation scheme was combined with the VBR-over-VBR approach
  - simulations were done to demonstrate the use of advanced resource management and routing methods
- Topics for future work
  - calculation of transient blocking probabilities by other methods
  - more accurate allocation function
  - robust allocation function
  - development of the VBR-over-VBR approach
  - study the cooperation of dynamic VPC capacity allocation and dynamic VCC routing