A High-Performance Computing Approach to Wireless Network Optimisation

Matthew C. Valenti

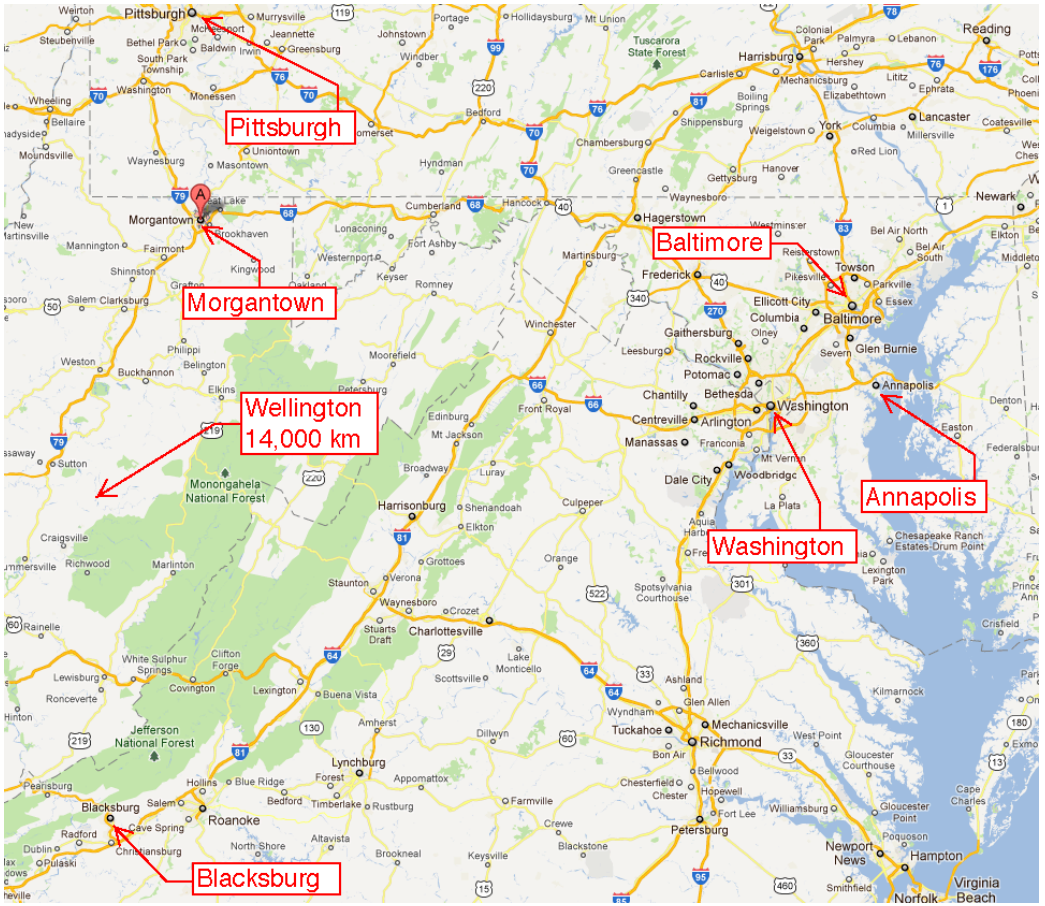
West Virginia University

Feb. 3, 2012

This work supported by the National Science Foundation under award CNS-0750821.

About Me

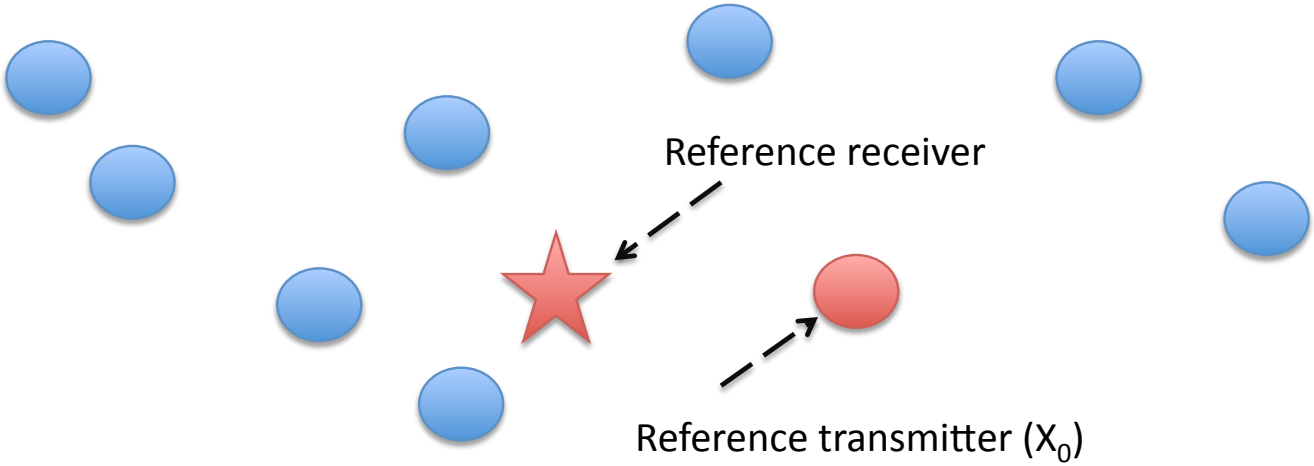
- From the state of Maryland, in the United States.
- Educated at
  - Virginia Tech
  - Johns Hopkins University.
- Worked as an Electronics Engineer at the U.S. Naval Research Laboratory.
- Professor at West Virginia University.



Outline

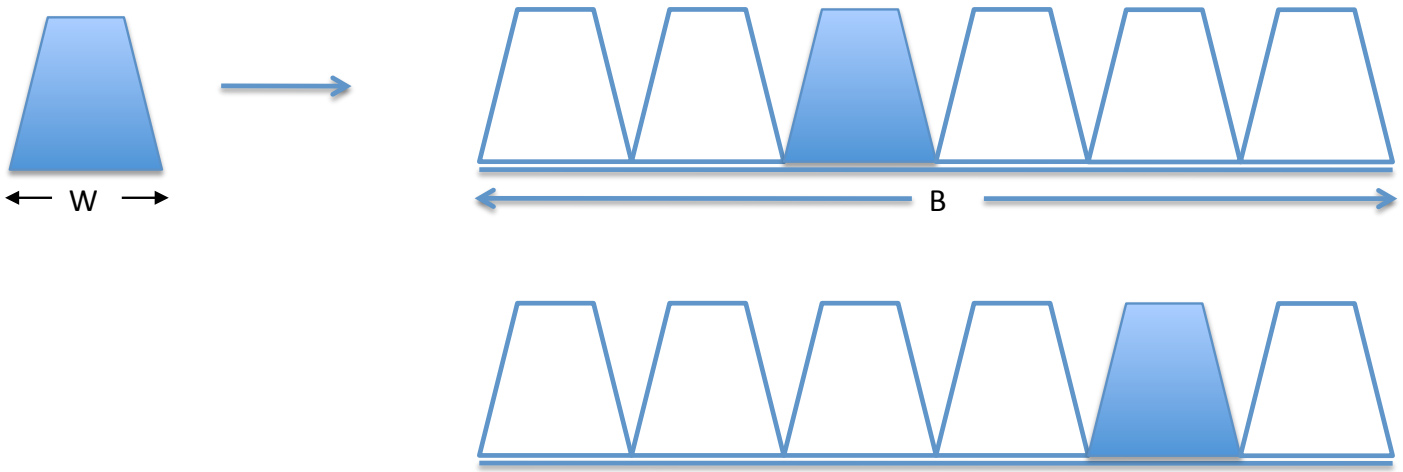
Outline

Ad Hoc Networks



- Transmitters are randomly placed in 2-D space.
  - X_i denotes 2-D location of i^{th} node.
  - Spatial model important (usually Poisson Point Process).
- Each node transmits to a random receiver.
  - Reference receiver located at the origin.
  - $|X_i|$ is distance to i^{th} node.
  - X_0 is location of reference transmitter.
  - M interfering transmitters, $\{X_1, ..., X_M\}$.

Frequency Hopping



- Transmitters randomly pick from among L frequencies.
- A reference receiver will pick same frequency as the reference transmitter with probability $p = 1/L$.
- Interference *avoidance*.
- Preferred for ad hoc networks.

SINR

The performance at the reference receiver is characterized by the signal-to-interference and noise ratio (SINR), given by:

$$\gamma = \frac{g_0 \Omega_0}{\Gamma^{-1} + \sum_{i=1}^{M} I_i g_i \Omega_i} \tag{1}$$

where:

- Γ is the SNR at unit distance.
- g_i is the power gain due to fading (i.e. Rayleigh or Nakagami fading).
- I_i is a Bernoulli indicator with $P[I_i] = p$.
- $\Omega_i = \frac{P_i}{P_0} 10^{\xi_i/10} ||X_i||^{-\alpha}$ is the normalized received power.
  - P_i is the power of transmitter X_i.
  - ξ_i is the dB shadowing gain (i.e. log-normal shadowing).
  - α is the path loss.

Outage Probability

- An *outage* occurs when $\gamma \leq \beta$, where β is an SINR threshold.
- From (1), the outage probability is

$$\begin{aligned}
\epsilon &= P\Big[\underbrace{\frac{g_0\Omega_0}{\Gamma^{-1}+\sum_{i=1}^{M} I_i g_i \Omega_i}}_{\gamma} \leq \beta\Big] \\
&= P\Big[\underbrace{\beta^{-1} g_0\Omega_0 - \sum_{i=1}^{M} I_i g_i \Omega_i}_{\mathsf{Z}} \leq \Gamma^{-1}\Big] \\
&= P\left[\mathsf{Z} \leq \Gamma^{-1}\right] = F_{\mathsf{Z}}(\Gamma^{-1}).
\end{aligned}$$

- To find the outage probability, we should find an expression for the cdf of Z.

Rayleigh Fading

- When all links are subject to Rayleigh fading,

$$F_{\mathsf{Z}}(z) \quad = \quad 1 - e^{-\beta z} \prod_{i=1}^{M} \frac{1 + \beta(1-p)\Omega_i}{1 + \beta\Omega_i}. \tag{2}$$

where it is assumed that:

- The reference transmitter is at unit distance, $|X_0| = 1$.
- There is no shadowing.

Nakagami Fading

If the channel from the i^{th} node to the receiver is Nakagami-m with parameter m_i, then for integer m_0,

$$
\begin{aligned}
F_{\mathsf{Z}}(z) &= 1 - \exp\left\{\beta z \frac{m_0}{\Omega_0}\right\} \sum_{s=0}^{m_0-1} \left(\beta z \frac{m_0}{\Omega_0}\right)^s \sum_{r=0}^{s} \frac{z^{-r} V_r(\mathbf{\Psi})}{(s-r)!} \\
V_r(\mathbf{\Psi}) &= \sum_{\substack{\ell_i \geq 0 \\ \sum_{i=0}^{M} \ell_i = r}} \prod_{i=1}^{M} U_{\ell_i}(\Psi_i) \\
U_\ell(\Psi_i) &= \begin{cases} 1 - p(1 - \Psi_i^{m_i}), & \text{for } \ell = 0 \\ \frac{p\Gamma(\ell+m_i)}{\ell!\Gamma(m_i)} \left(\frac{\Omega_i}{m_i}\right)^\ell \Psi_i^{m_i+\ell}, & \text{for } \ell > 0 \end{cases} \\
\Psi_i &= \left(\beta \left(\frac{m_0}{\Omega_0}\right) \left(\frac{\Omega_i}{m_i}\right) + 1\right)^{-1}, \qquad \text{for } i = \{1, ..., M\}.
\end{aligned}
$$

An Example

- Reference (source) transmitter placed at distance $|X_0| = 1$.
- $M = 50$ interferers randomly placed in a circle of radius $r_{max} = 4$.
- $L = 200$ hopping frequencies, i.e. $p = 1/200 = 0.005$.
- $\beta = 3.7$ dB SINR threshold.
- Three fading models considered:
  - **Rayleigh fading:** $m_i = 1$ for all i.
  - **Nakagami fading:** $m_i = 4$ for all i.
  - **Mixed fading:** $m_0 = 4$ for source and $m_i = 1$ for interferers.
- Path-loss coefficient $\alpha = 3$.
- No shadowing.

Example #1

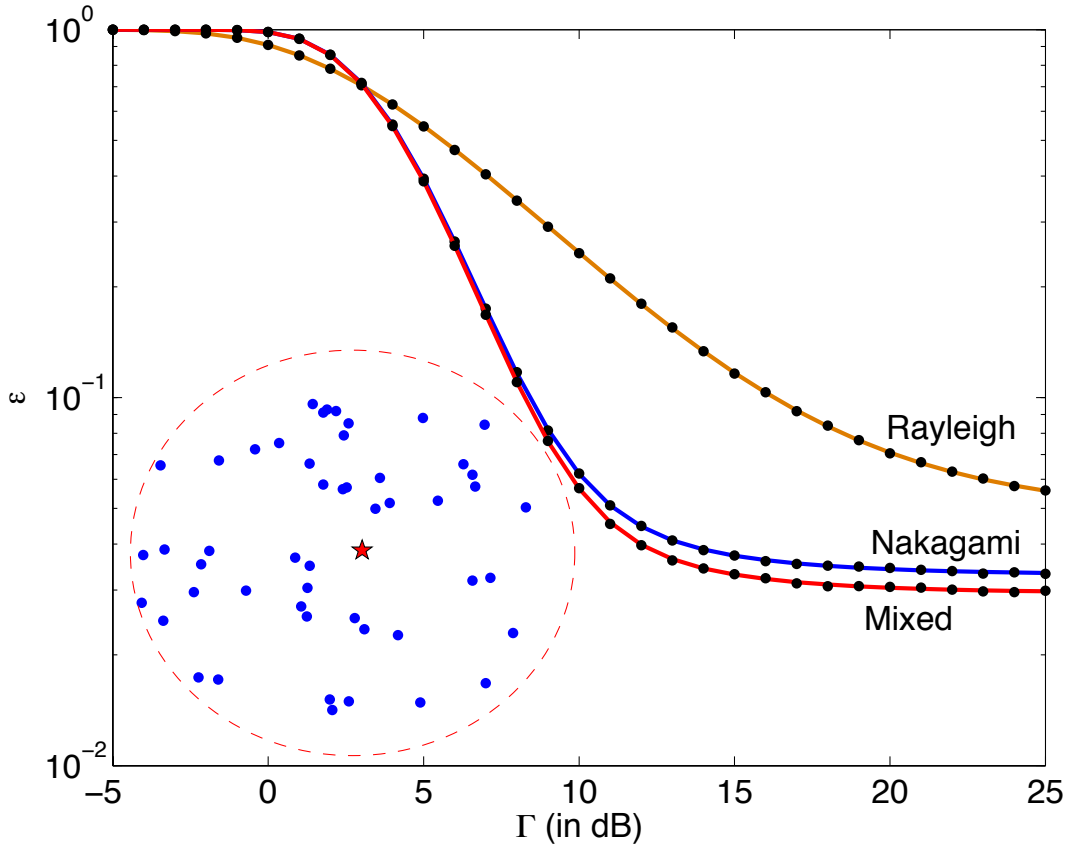


Figure: Outage probability ϵ as a function of SNR Γ. Analytical curves are solid, while • represents simulated values. The network geometry is shown in the inset, with the reference receiver represented by ⋆ and interferers by •.

Spatial Averaging

- ϵ is the outage probability conditioned on a specific network topology.
- We may want to average the outage probability with respect to the spatial distribution.
- The averaging can be done in closed-form for some, but not all, cases.
- When a closed-form solution does not exist, a Monte Carlo approach can be taken:
  - Draw N networks, each of size M.
  - Let $\mathbf{\Omega}_j$ be the set of Ω_i's for the j^{th} network.
  - Let $F_{\mathsf{Z}}(z|\mathbf{\Omega}_j)$ be the cdf of Z for the j^{th} network.
  - Take the average of the N cdfs

$$F_{\mathsf{Z}}(z) \quad = \quad \frac{1}{N}\sum_{j=1}^{N} F_{\mathsf{Z}}(z|\mathbf{\Omega}_j).$$

  - As before, the outage probability is $\epsilon = F_{\mathsf{Z}}(\Gamma^{-1})$.
- **Shadowing** can be modeled by including the factor $10^{\xi_i/10}$ in each Ω_i. For log-normal shadowing ξ_i is zero mean Gaussian with variance σ_s^2.

Spatially-Averaged Outage

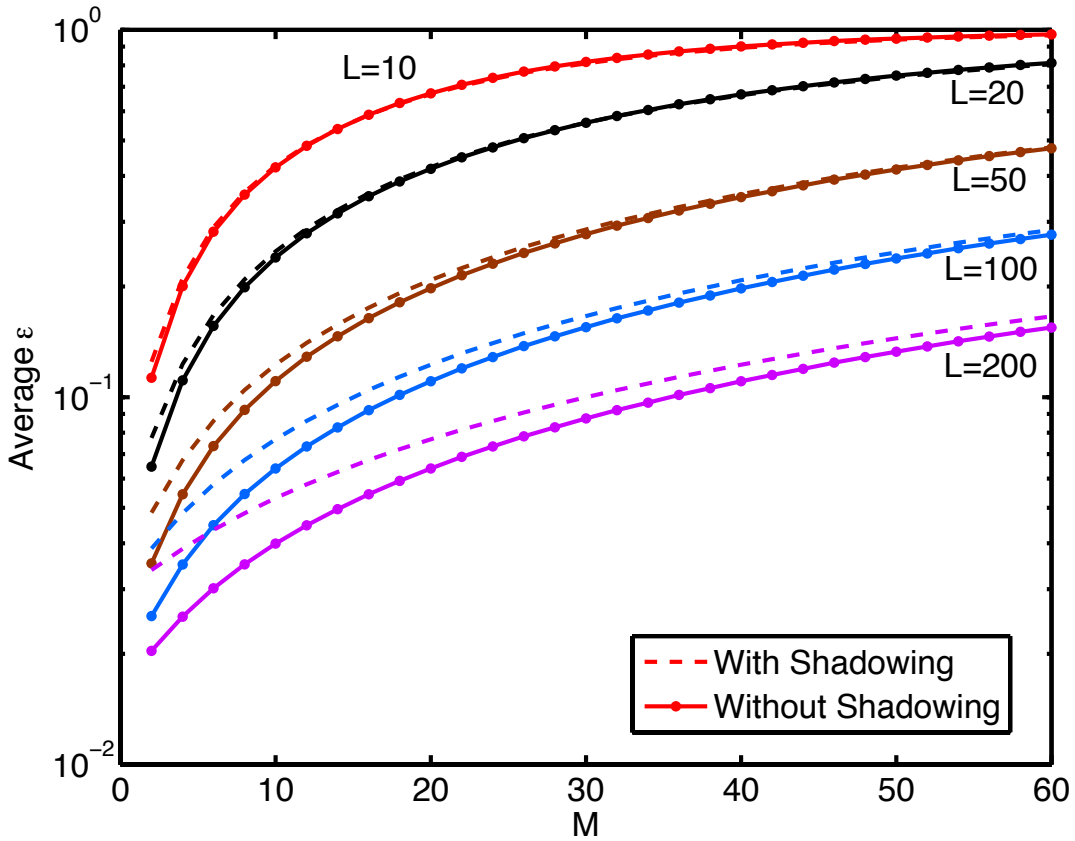


Figure: Average outage probability as function of the number of interferers M at SNR $\Gamma = 10$ dB with SINR threshold $\beta = 3.7$ dB in a mixed-fading environment. Curves are shown both with ($\sigma_s^2 = 8$) and without shadowing. Results were obtained by averaging over $N = 10\,000$ randomly generated networks.

SINR Threshold

- Until now, we have picked the SINR threshold β arbitrarily.
- β depends on the choice of modulation.
  - For *ideal* signaling

$$C(\gamma) = \log_2(1+\gamma)$$

β is the value of γ for which $C(\gamma) = R$ (the code rate),

$$\beta = 2^R - 1$$

  - For other modulations, the *modulation-constrained* capacity must be used.

Modulation for Frequency Hopping

$$s_d(t) = \frac{1}{\sqrt{T_s}} e^{j2\pi dt/T_s}, \ d = 0, 1, \cdots, q-1$$

- Orthogonal FSK
  - Suitable for noncoherent reception.
  - Reasonable energy efficiency.
  - Poor bandwidth efficiency because adjacent tones are $1/T_s$ apart.
- Nonorthogonal CPFSK
  - Reduce bandwidth by using modulation index $h < 1$.
  - Adjacent frequency tones are h/T_s apart.
  - Continuous-phase constraint controls the spectrum.
  - Transmitted $x(t) = e^{j\phi} s_d(t)$ where phase ϕ is accumulated

$$\phi = \phi' + 2\pi dh$$

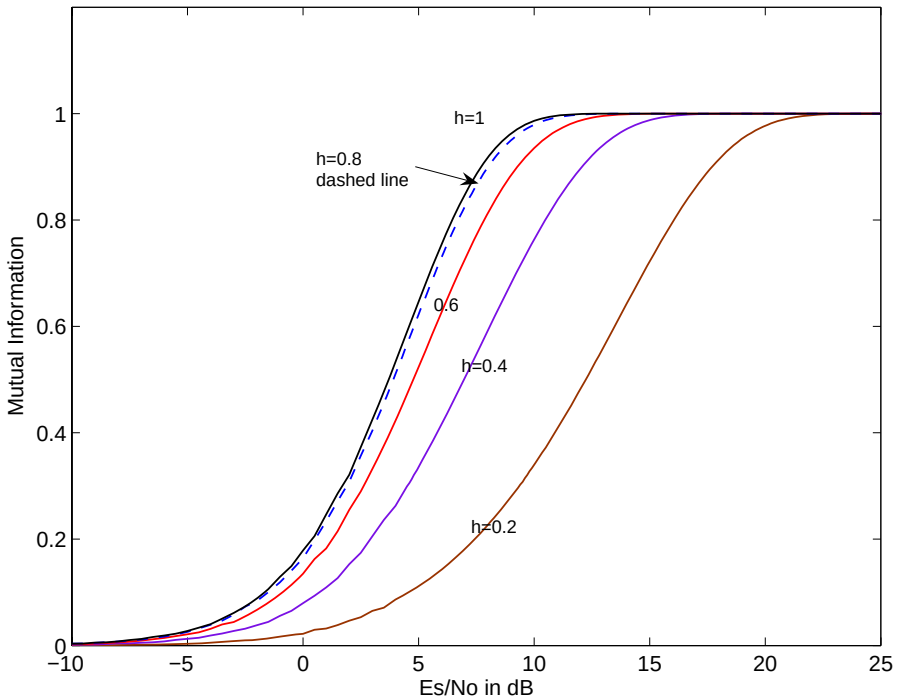
Modulation for Frequency Hopping

$$s_d(t) = \frac{1}{\sqrt{T_s}} e^{j2\pi dht/T_s}, \ d = 0, 1, \cdots, q-1$$

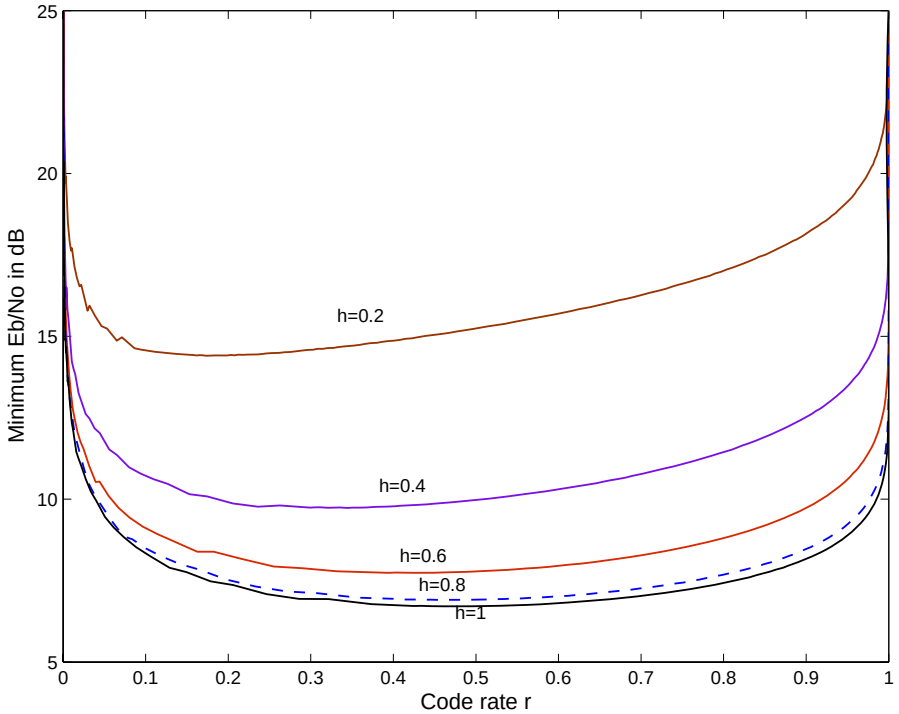
- Orthogonal FSK
  - Suitable for noncoherent reception.
  - Reasonable energy efficiency.
  - Poor bandwidth efficiency because adjacent tones are $1/T_s$ apart.
- Nonorthogonal CPFSK
  - Reduce bandwidth by using modulation index $h < 1$.
  - Adjacent frequency tones are h/T_s apart.
  - Continuous-phase constraint controls the spectrum.
  - Transmitted $x(t) = e^{j\phi} s_d(t)$ where phase ϕ is accumulated

$$\phi = \phi' + 2\pi dh$$

Capacity of Noncoherent Binary CPFSK



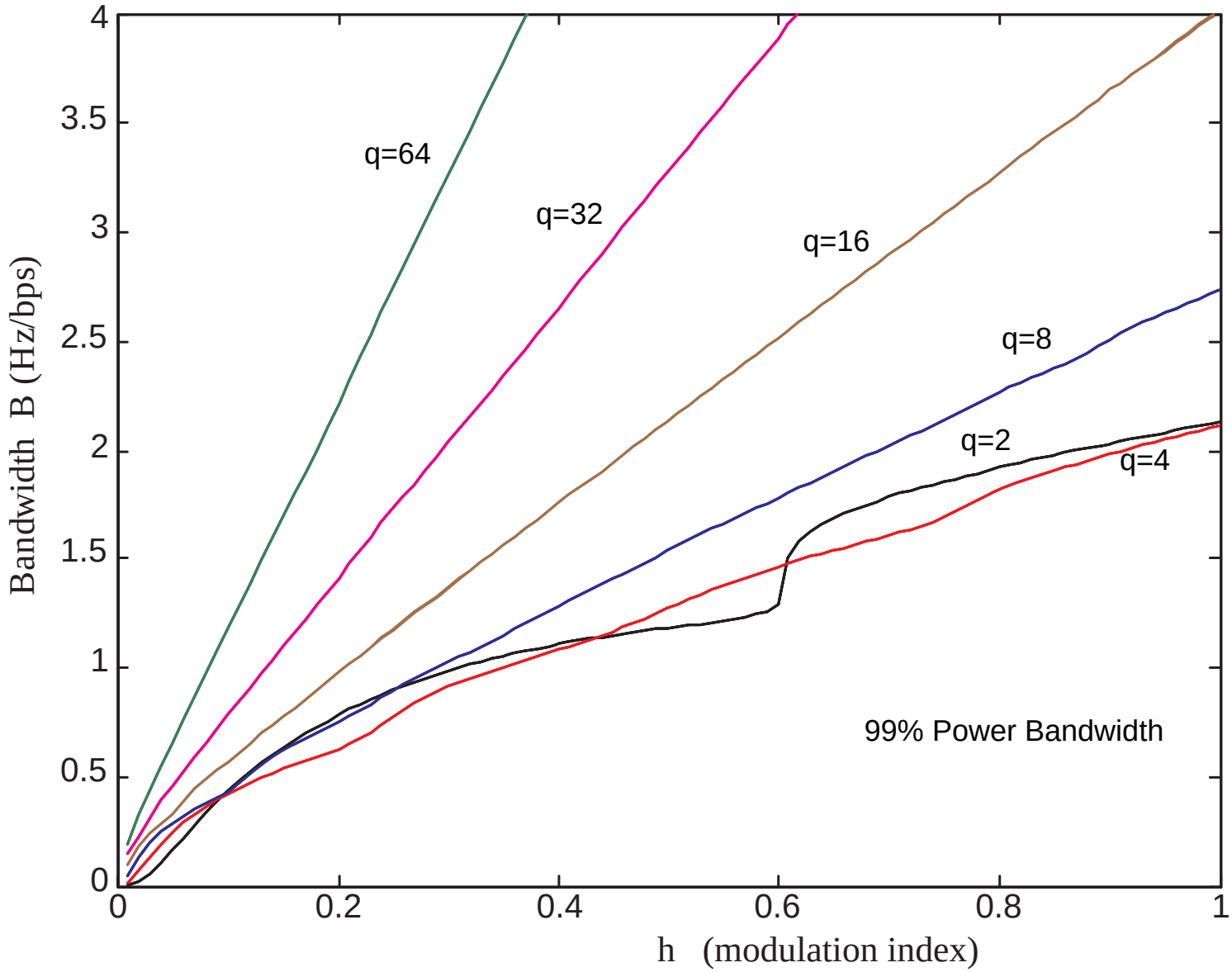
(a) channel capacity versus $\mathcal{E}_S/N_0$



(b) minimum $\mathcal{E}_b/N_0$ versus coding rate

Reference: S. Cheng, R. Iyer Sehshadri, M.C. Valenti, and D. Torrieri, "The capacity of noncoherent continuous-phase frequency shift keying," in *Proc. Conf. on Info. Sci. and Sys. (CISS)*, (Baltimore, MD), Mar. 2007.

Bandwidth of CPFSK



The Network Optimisation Problem

- The objective is to maximize the *transmission capacity*, which for FH is

$$\tau = \frac{\eta R}{L}(1-\epsilon)\lambda$$

  where
  - η the (uncoded) modulation's spectral efficiency (bps/Hz).
  - R is the rate of the channel code.
  - L is the number of hopping frequencies.
  - $\lambda = M/A$ is the *density* of the network, where A is the *area* of the network.
  - ϵ is the spatially-averaged outage probability.
- Transport capacity is the area spectral efficiency.
  - Units of bps/Hz/m^2.
  - The rate that bits are successfully transmitted over 1 Hz BW and 1 square meter of area.
- Goal of the optimisation is to pick R, h, and L that maximize τ.

Optimisation Algorithm

Brute force optimisation, assuming constant transmit power $P_i = P_0, \forall i$,

1. Create a set of $\{\mathbf{\Omega}_j\}$ corresponding to M networks, where each $\mathbf{\Omega}_j$ is found by placing N nodes $\{X_i\}$ according to the desired spatial distribution and then computing $\Omega_i = 10^{\xi_i/10}||X_i||^{-\alpha}$ for each node, where ξ_i is drawn from the desired shadowing distribution.
2. For each L and β in a discretized set, perform the following **task**:
   1. Compute the outage probability ϵ averaged over the $\{\mathbf{\Omega}_j\}$.
   2. For each h in a discretized set, determine the rate $R = C(\beta)$, which is the achievable rate of noncoherent CPFSK with modulation index h and SNR β.
   3. For the set of (h, R) found in the last step, determine the normalized transmission capacity $\tau = \eta R \lambda (1 - \epsilon)/L$.
3. Identify the (h, R, L) that maximizes τ.
4. If desired, adjust the spacing and range of (h, R, L) and return to 2.

Example Optimisation

- Nakagami parameters: $m_0 = 4$ and $m_i = 1$ (mixed fading).
- Shadowing variance: $\sigma_s^2 = 8$.
- Path-loss exponent: $\alpha = 3$.
- Number of interferers: $M = 50$.
- Network dimensions: $r_{min} = 0.25$ and $r_{max} = 2$.
- Number of networks generated: $N = 100$.

Influence of h

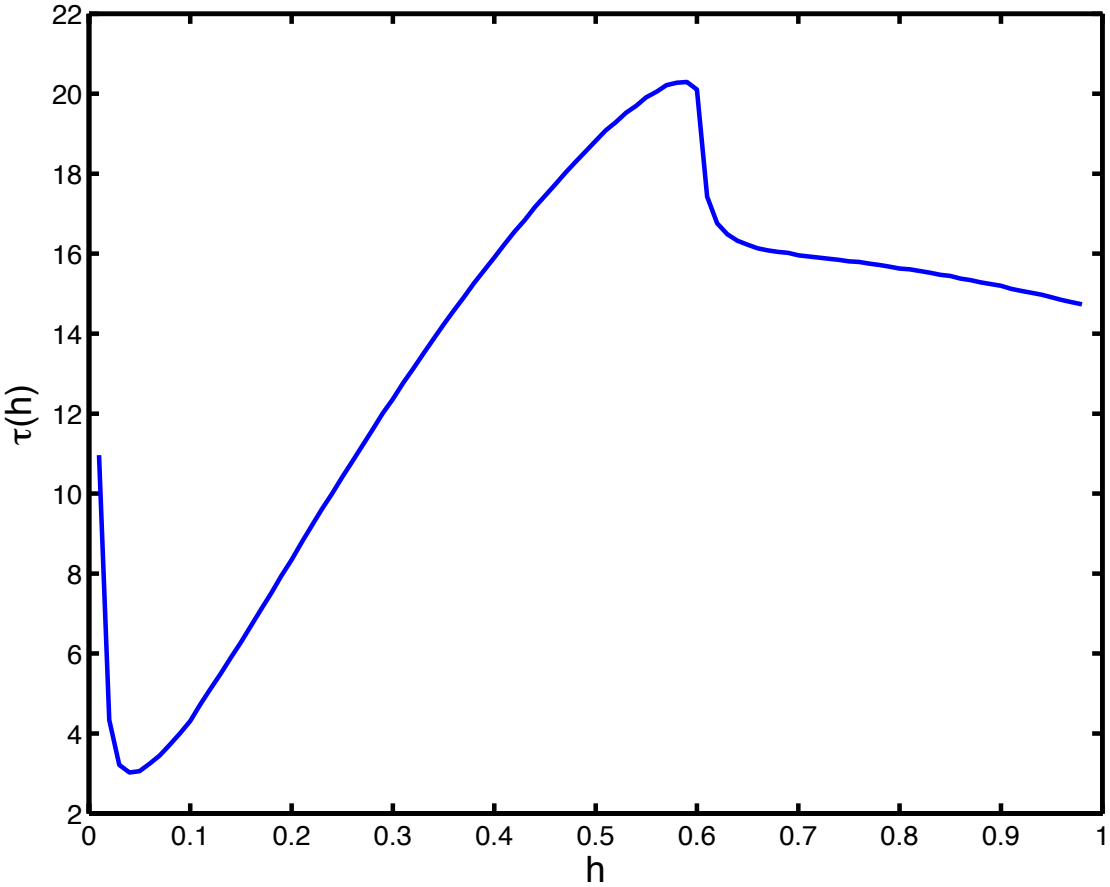


Figure: Maximum transmission capacity $\tau(h)$ (in $bps/kHz - m^2$) as a function of the modulation-index h. For each value of h, the code rate R and number of frequency channels L are varied to maximize the TC.

Influence of R

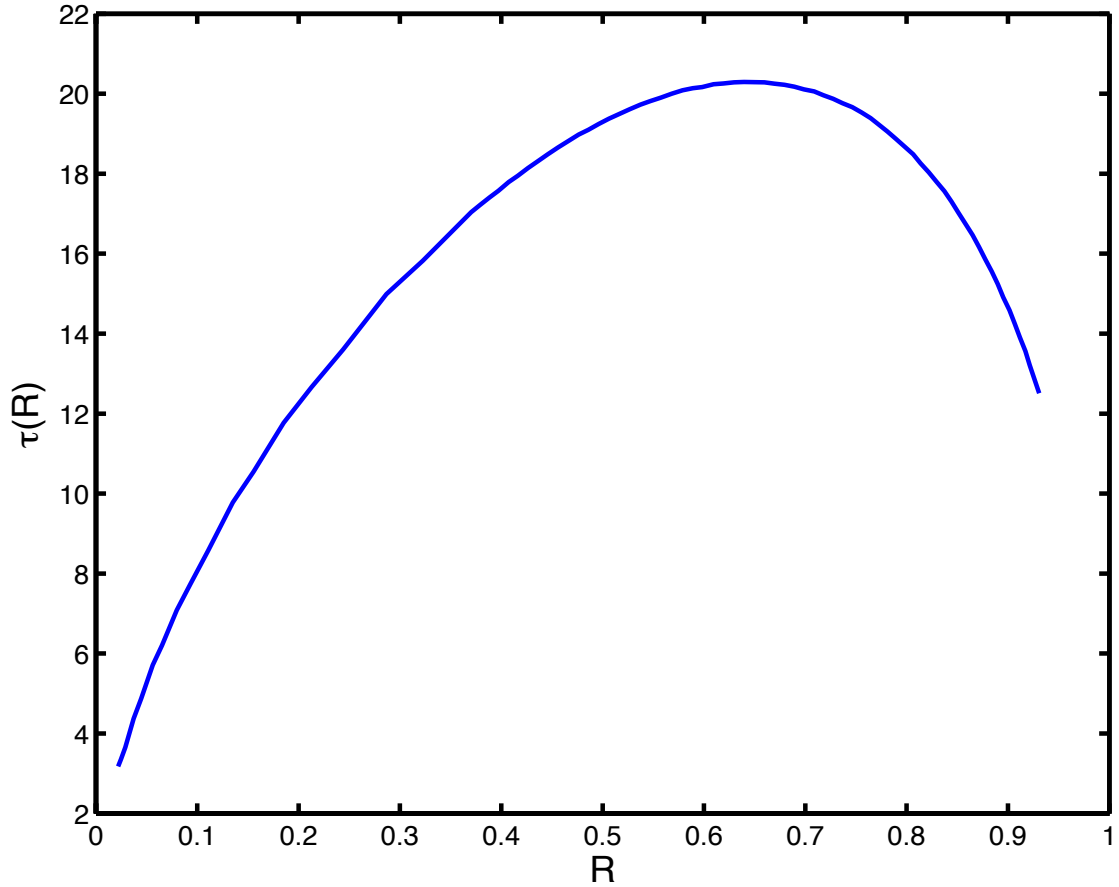


Figure: Maximum transmission capacity $\tau(R)$ (in $bps/kHz - m^2$) as a function of the code rate R. For each value of R, the modulation-index h and number of frequency channels L are varied to maximize the TC.

Influence of L

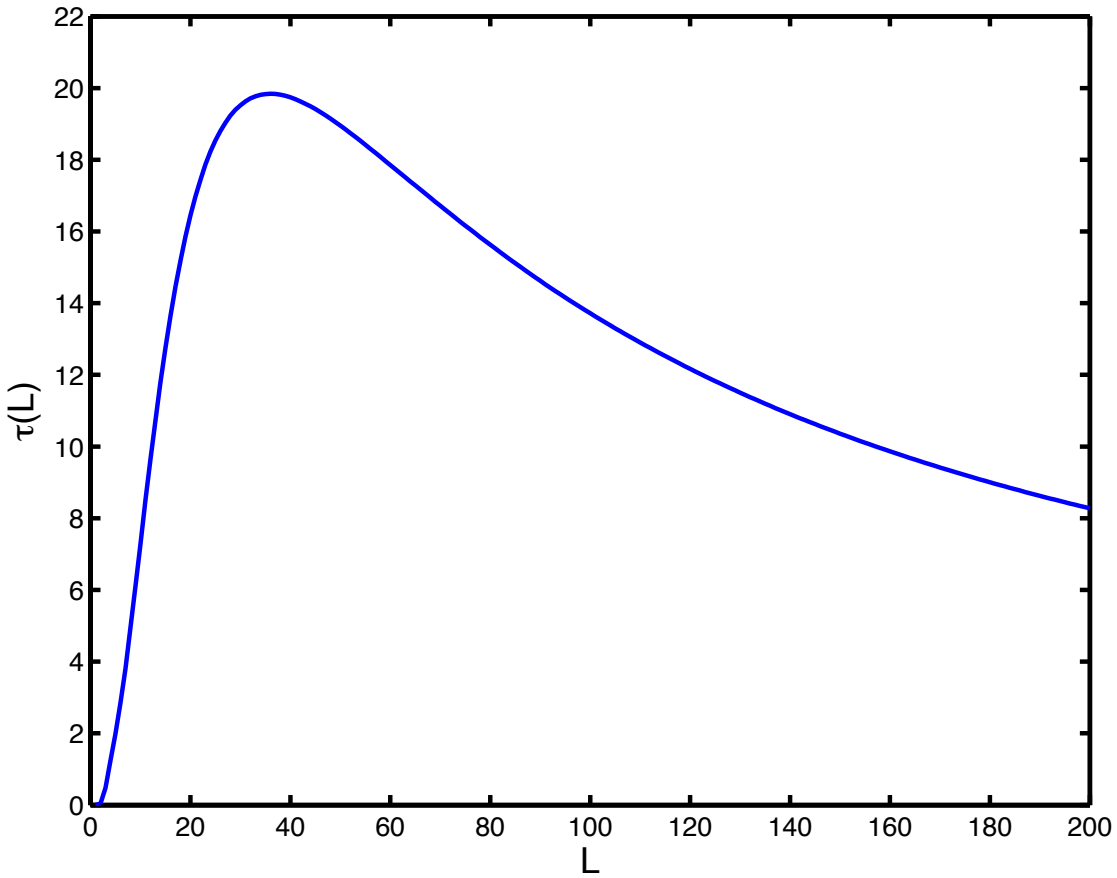


Figure: Maximum transmission capacity $\tau(L)$ (in $bps/kHz - m^2$) as a function of the number of frequency channels L. For each value of L, the modulation-index h and code rate R are varied to maximize the TC.

Optimisation Results

r_{max}	σ_s^2	m_0	m_i	L	R	h	τ_{opt}	τ_{sub}
2	0	1	1	31	0.61	0.59	15.92	3.31
		4	4	42	0.66	0.59	17.09	4.05
		4	1	36	0.65	0.59	19.82	4.13
	8	1	1	31	0.63	0.59	15.98	3.31
		4	4	41	0.66	0.59	17.43	4.04
		4	1	36	0.66	0.59	20.11	4.12
4	0	1	1	12	0.54	0.59	9.73	0.89
		4	4	15	0.50	0.59	10.65	1.12
		4	1	14	0.51	0.59	11.85	1.12
	8	1	1	12	0.53	0.59	9.41	0.89
		4	4	16	0.51	0.59	10.26	1.12
		4	1	14	0.52	0.59	11.46	1.12

Table: Results of the optimisation for $M = 50$ interferers. The transmission capacity τ is in units of bps/kHz-m^2. τ_{opt} is TC with the optimizer parameters, while τ_{sub} is TC with $(L, R, h) = (200, 1/2, 1)$.

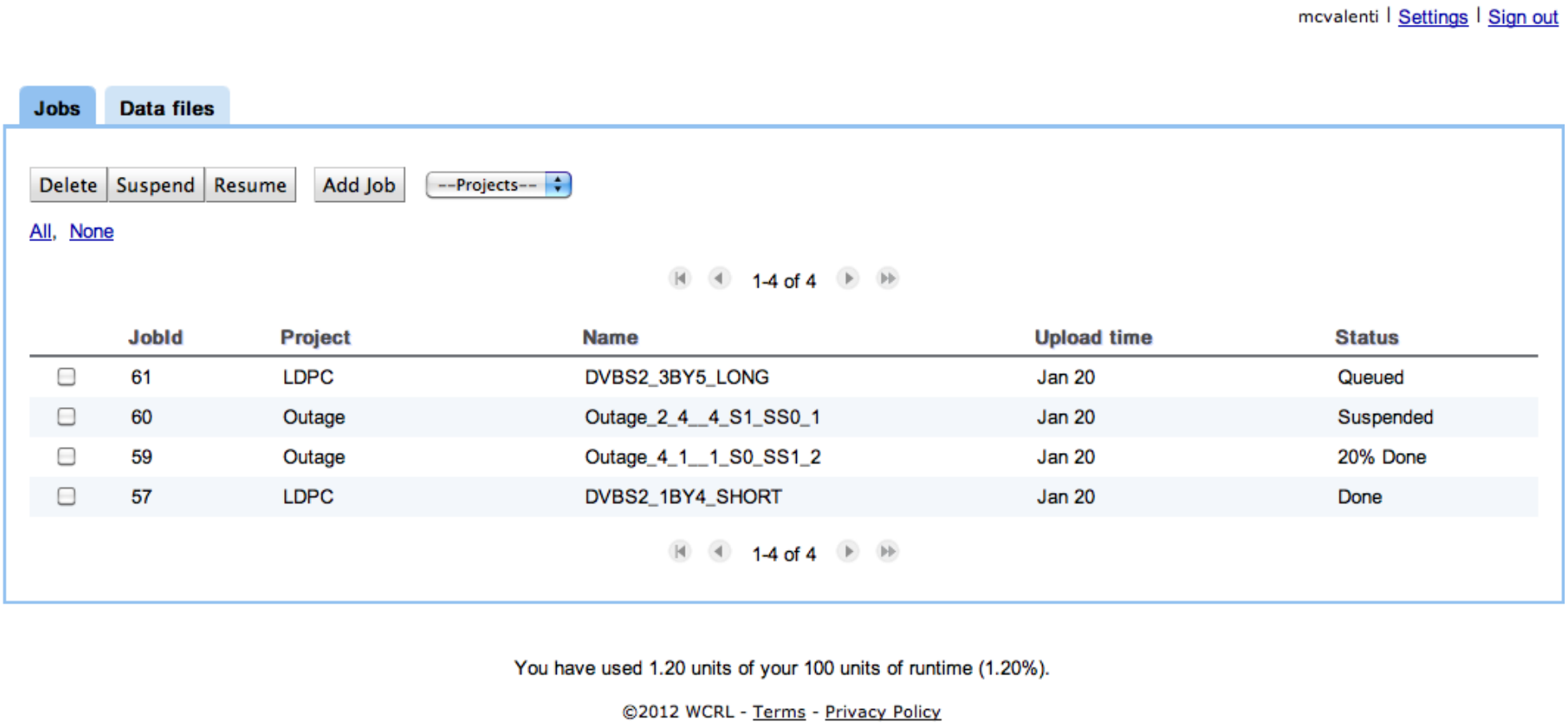
Outline

A Computing Cloud for Communication Theory

- Under NSF funding, we have created a cloud-computing resource to support the specific needs of the communication theory research community.
- The goals of our project are to:
  - Be compatible with Matlab.
  - Provide $100 - 1000$ times speedup relative to a single PC.
  - Be accessible by the research community through a web-interface.
  - Support multiple open-source *projects*, with code contributed from the CT community [1].
  - Allow the CT community to *donate* idle CPU cycles through the use of *volunteer computing*.
- Our philosophy is borrowed from Seymour Cray,
  "My guiding principle has always been simplicity. Only include features which are absolutely necessary."

[1] Google code page: http://code.google.com/p/iscml/

The User Interface



The web interface is developed using the **Google Web Toolkit** (GWT), which is the same technology used to implement *gmail*.

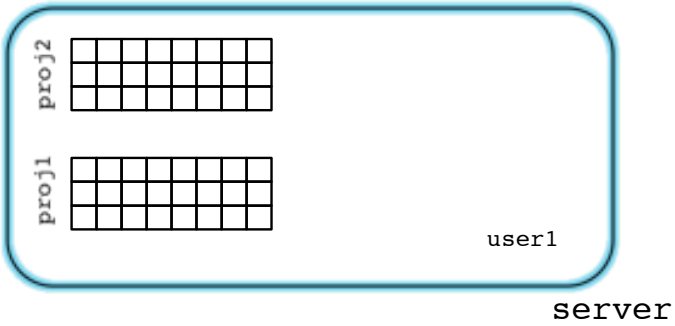
---

http://wcrlcluster.csee.wvu.edu:8180/ldpc/

Some Terminology

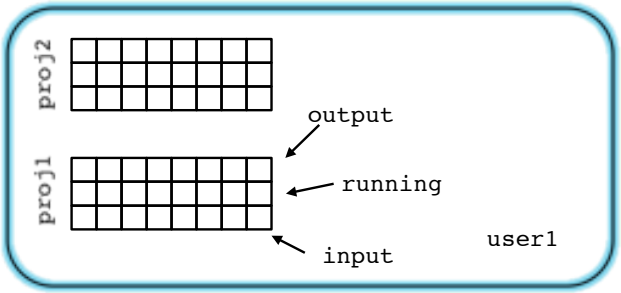
- A **project** is a collection of open-source code for solving a specific CT problem, e.g. simulate a LDPC code, optimize a wireless network.
- A **job** is a project-specific request to generate results, e.g. generate a BER curve, determine optimal network parameters. The request is embodied by a **job file**.
- A **task** is a a work-unit associated with a job that can be run on a single processor in a small increment of time (e.g., 5 minutes). The task is embodied by a **task file**.
- A **worker** is a process running on a computing cluster or grid that services tasks.
- A **queue** is an entity that holds job or task files that are either waiting to be serviced (input queue), being serviced (running queue), or completed (output queue).

How It Works



The user has a job queue
on the server for each project

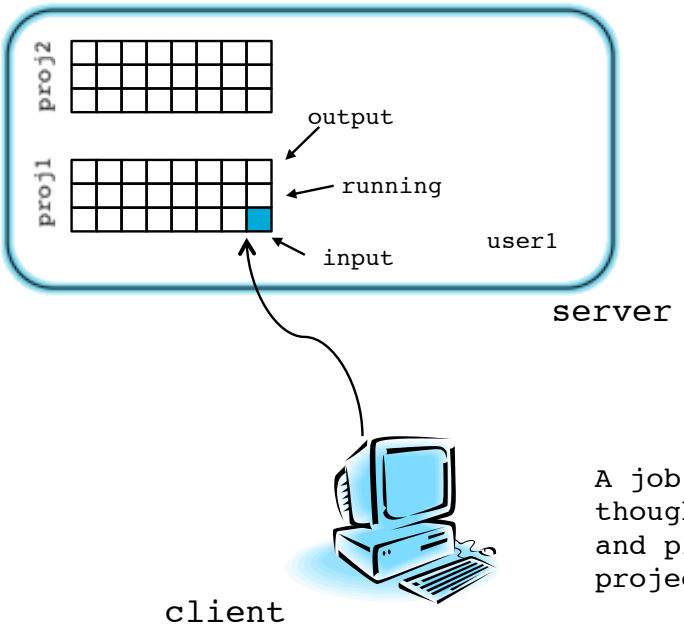
How It Works



The user has a job queue
on the server for each project.

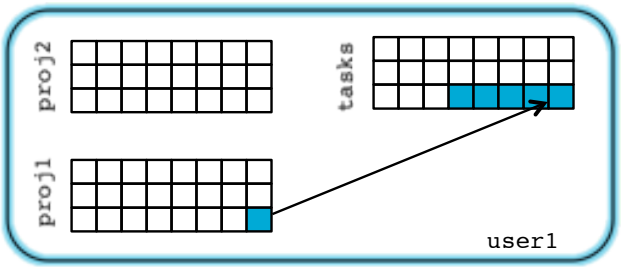
Each job queue has three components:
input, running, and output.

How It Works



A job file is uploaded
though the web interface
and placed in the
project's job input queue

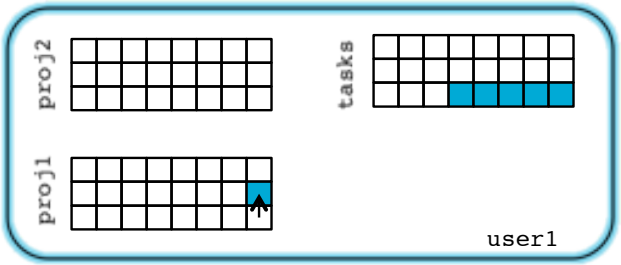
How It Works



The project-specific ***job manager*** places several task files into the user's *task input* queue

server

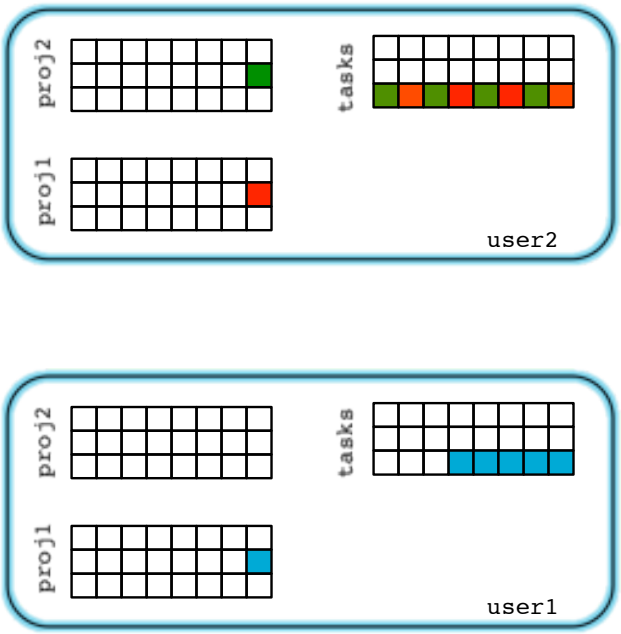
How It Works



The project-specific ***job manager*** places several task files into the user's *task input* queue, and moves the job file into the project's *job running* queue.

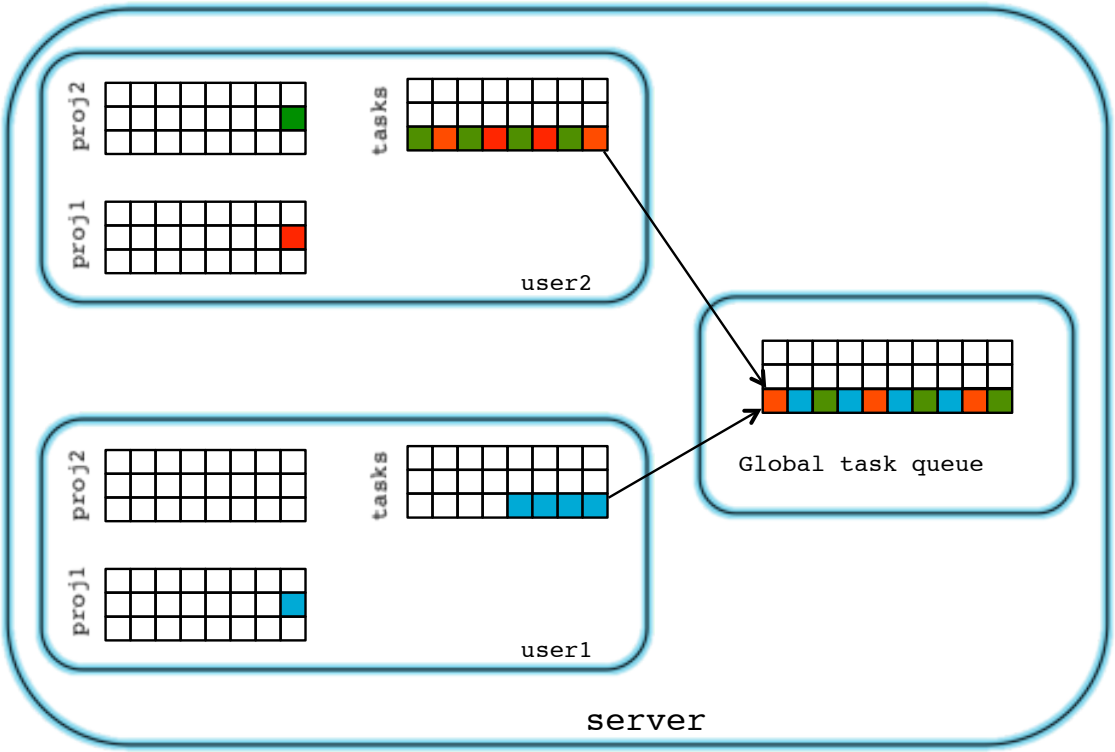
server

How It Works



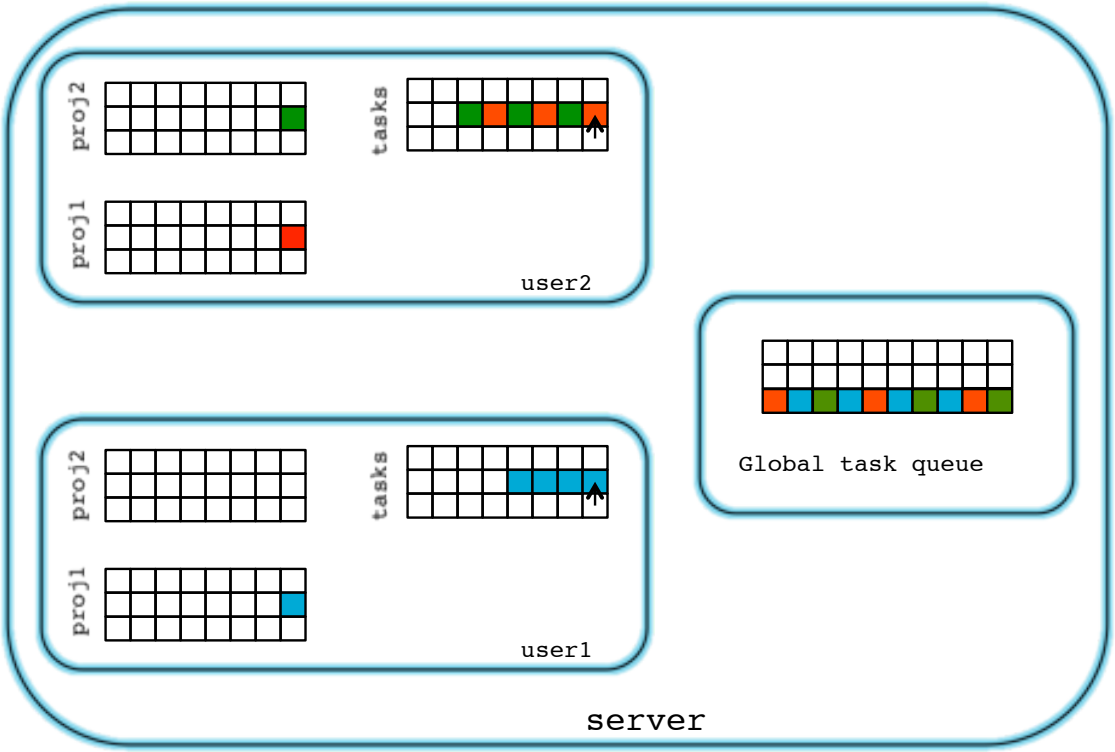
The system can support multiple users. Each user has a set of job queues for each project it is subscribed to, and a set of task queues.

How It Works



A ***task manager*** copies task files to the global *task input* queue.

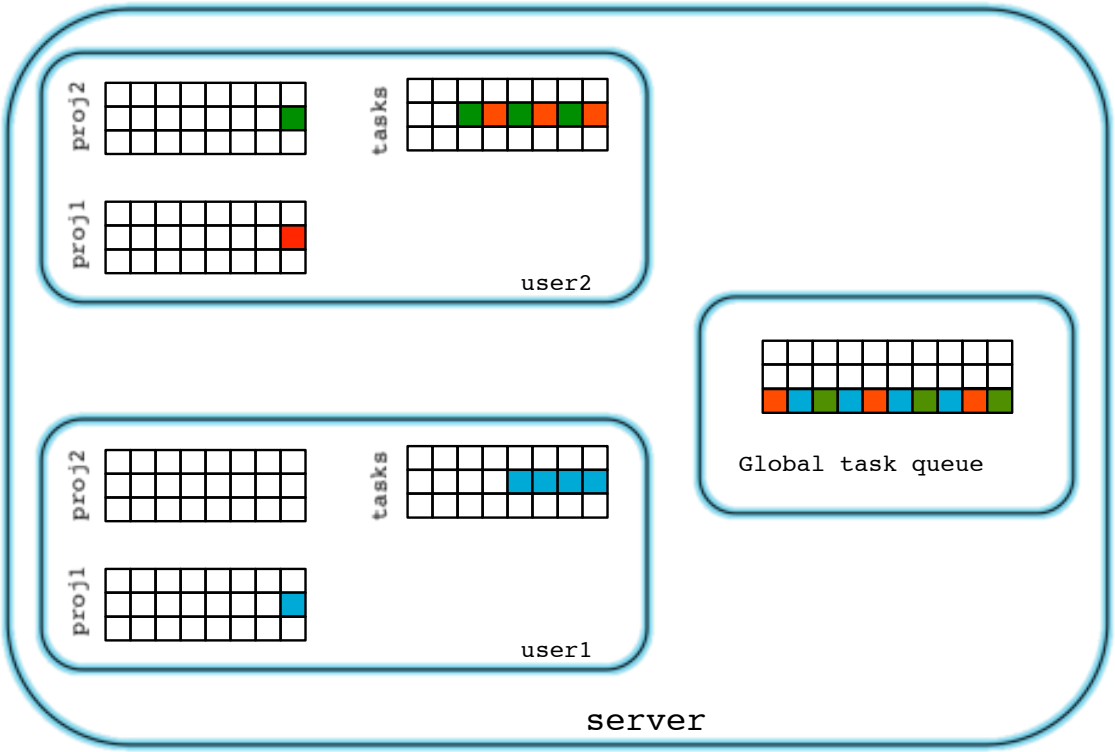
How It Works

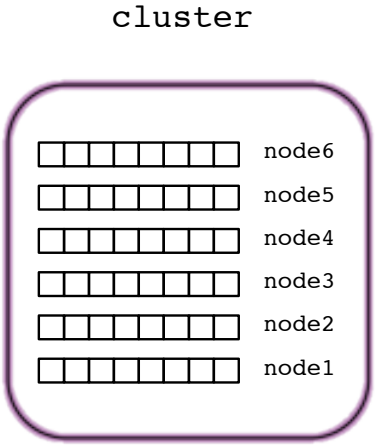


A ***task manager*** copies task files to the global *task input* queue.

The task files are moved to the user's *running task* queue.

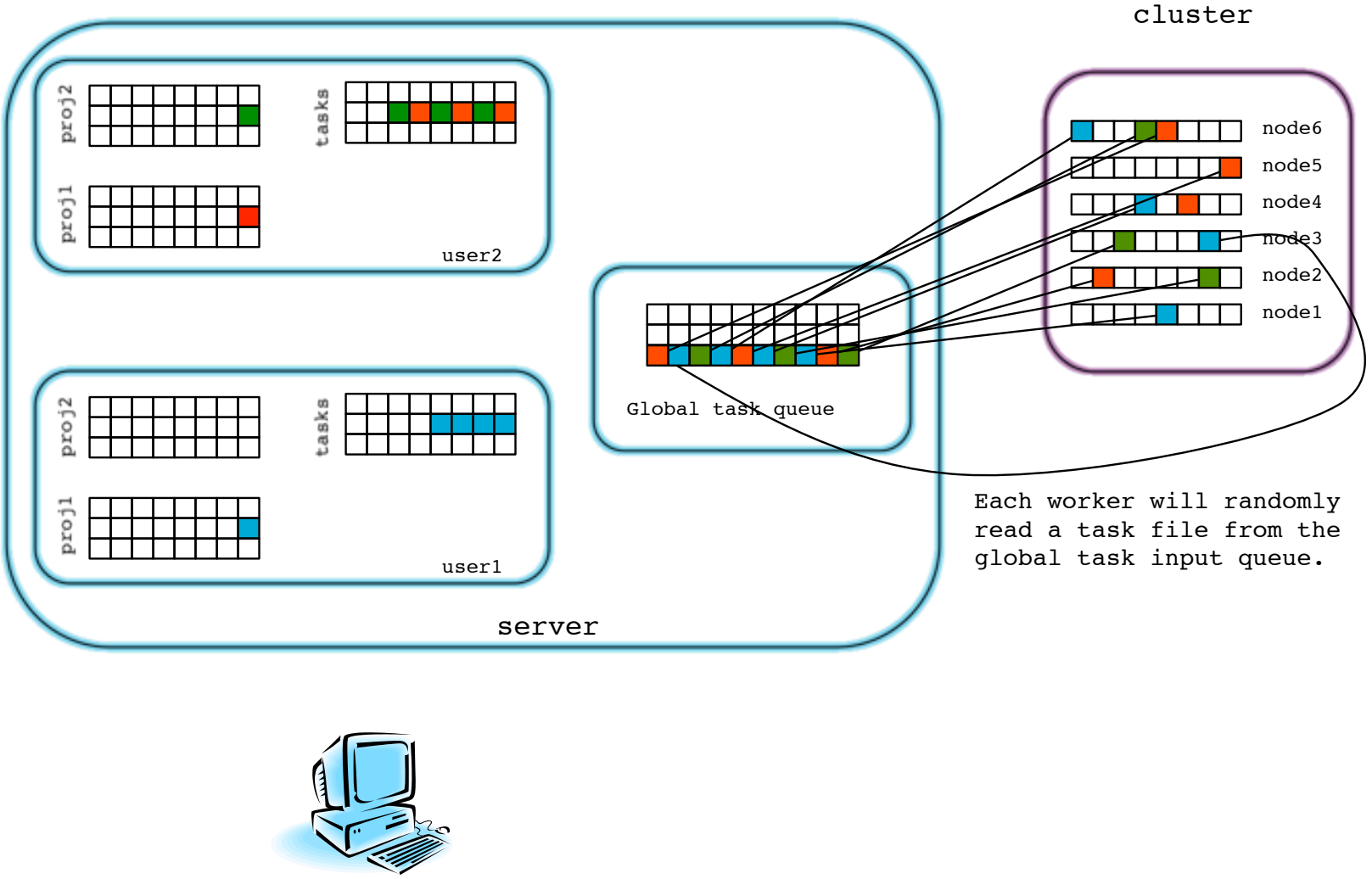
How It Works



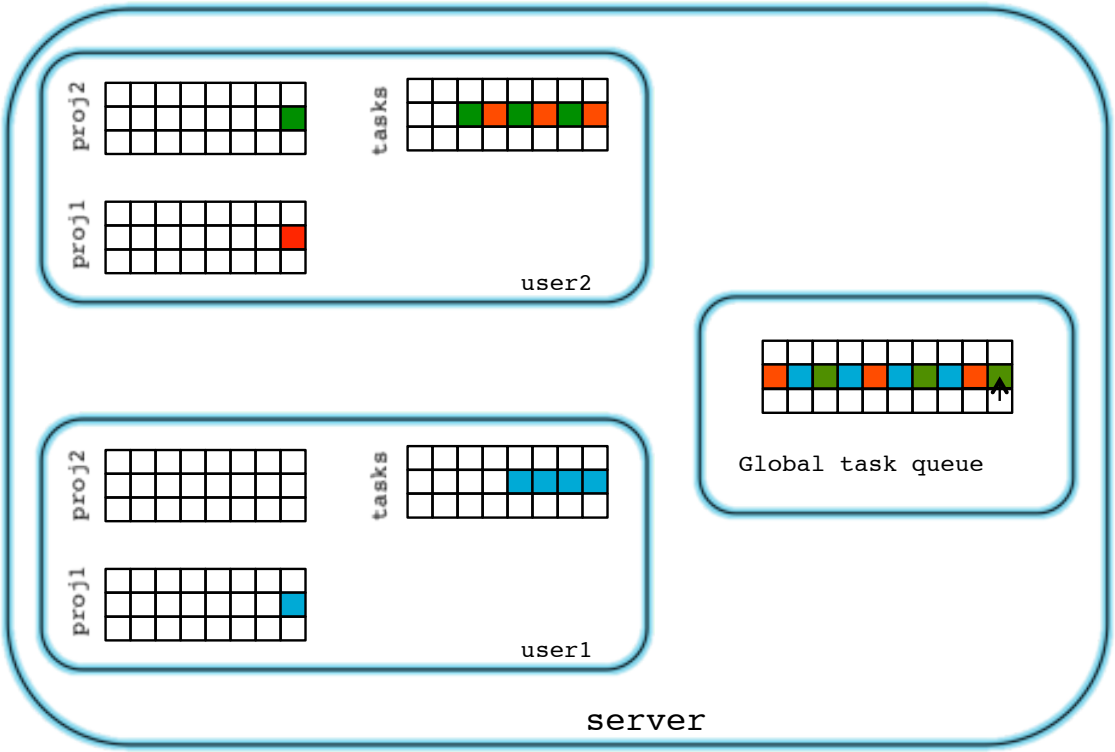


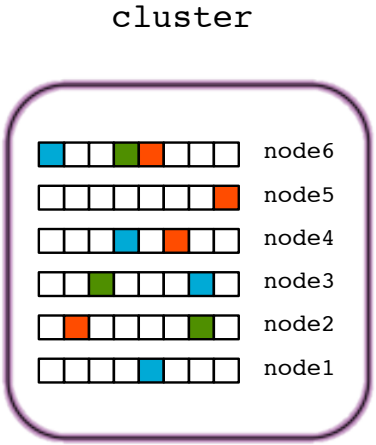
The cluster has several *workers* running on each node.

How It Works


cluster
proj2
tasks
proj1
user2
proj2
tasks
proj1
user1
server
Global task queue
node6
node5
node4
node3
node2
node1
Each worker will randomly
read a task file from the
global task input queue.

How It Works

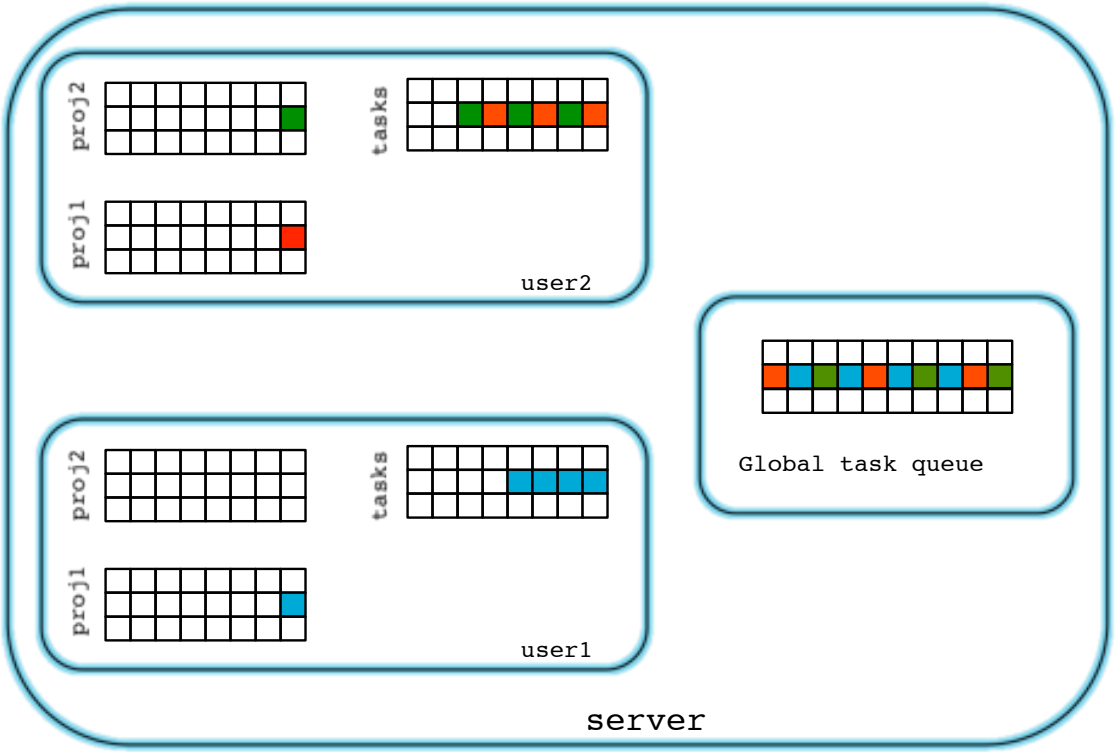


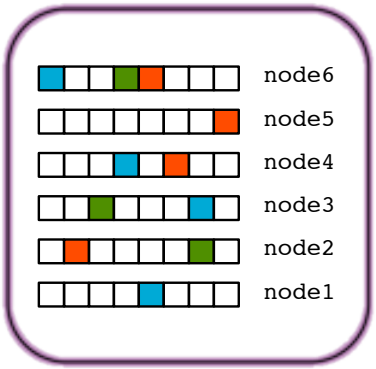


Each worker will randomly read a task file from the global task input queue.

The workers move the task files to the global *running* queue.

How It Works



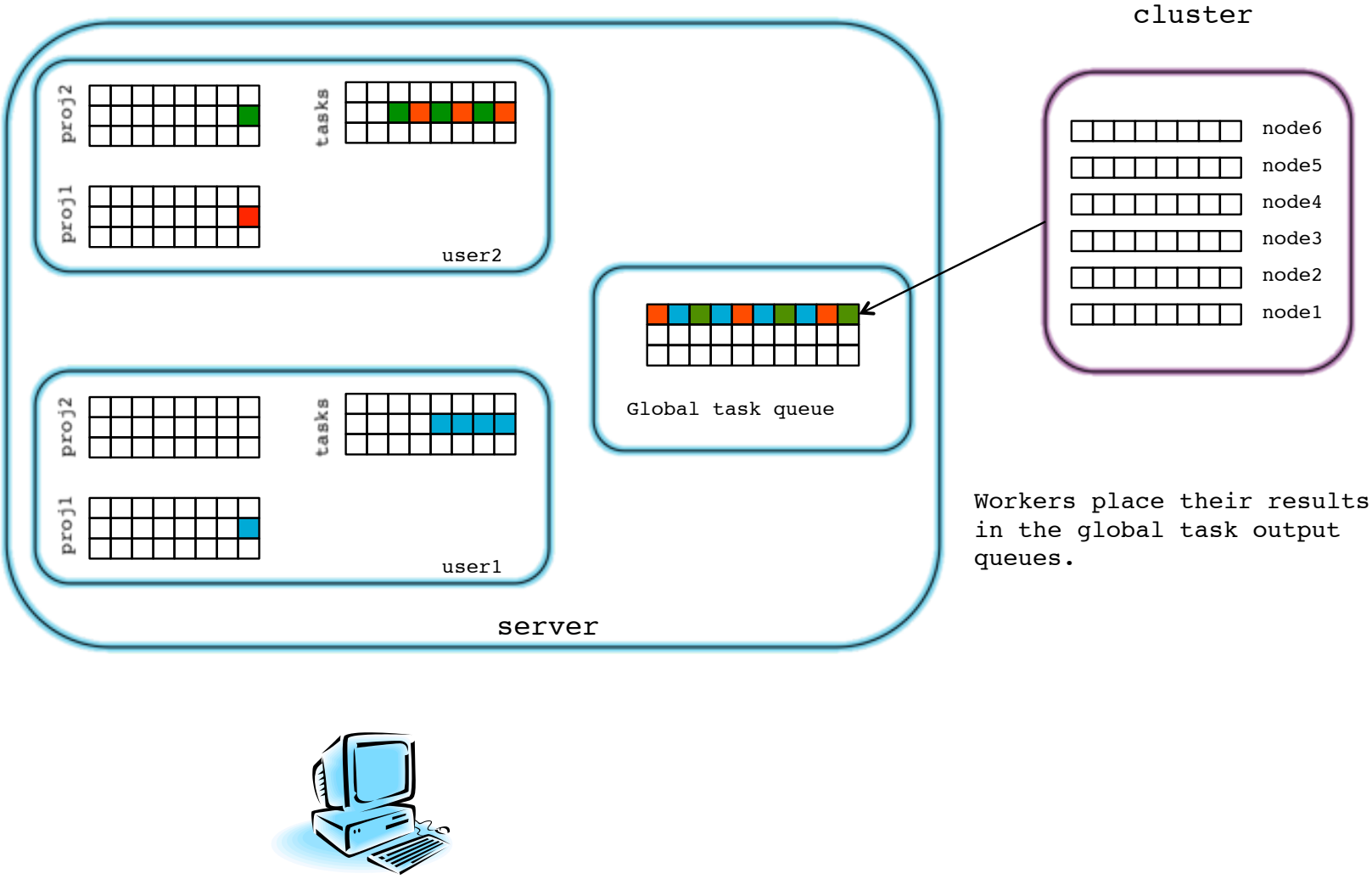


Each worker will randomly pick a task file from the global task input queue.

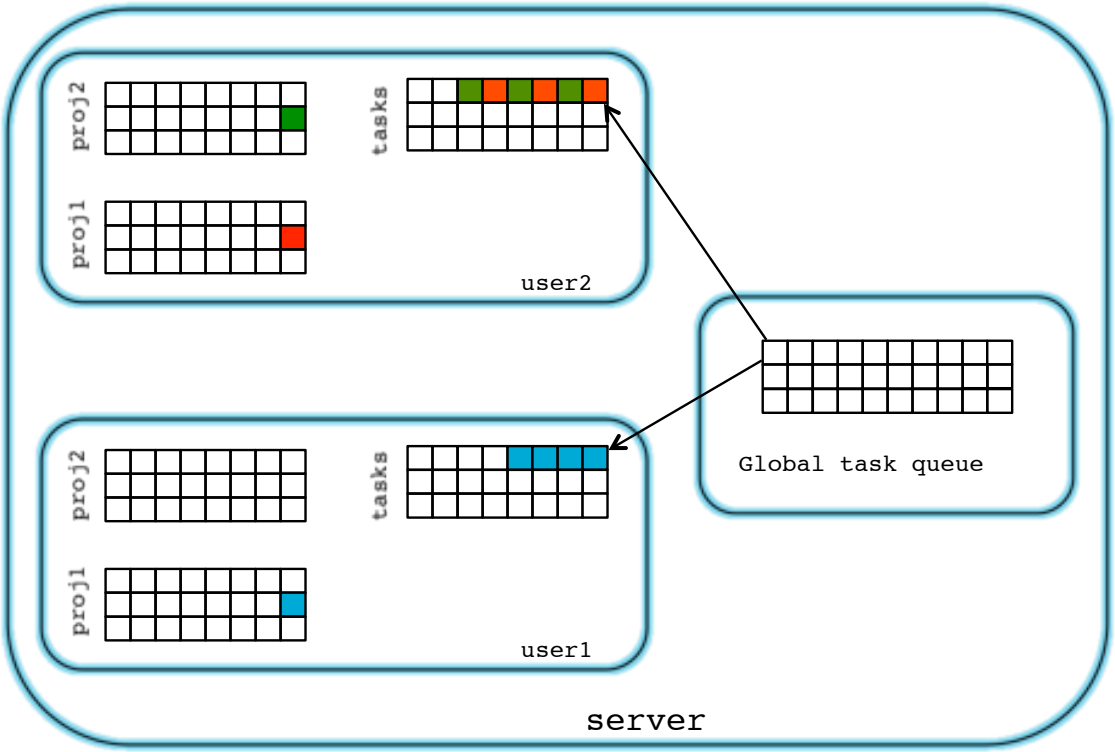
The workers move the task files to the global *running* queue.

Each worker runs its task to completion. This typically takes 5 minutes.

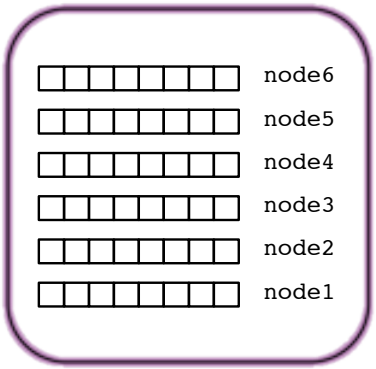
How It Works


cluster
node6
node5
node4
node3
node2
node1
proj2
tasks
proj1
user2
Global task queue
proj2
tasks
proj1
user1
server
Workers place their results in the global task output queues.

How It Works

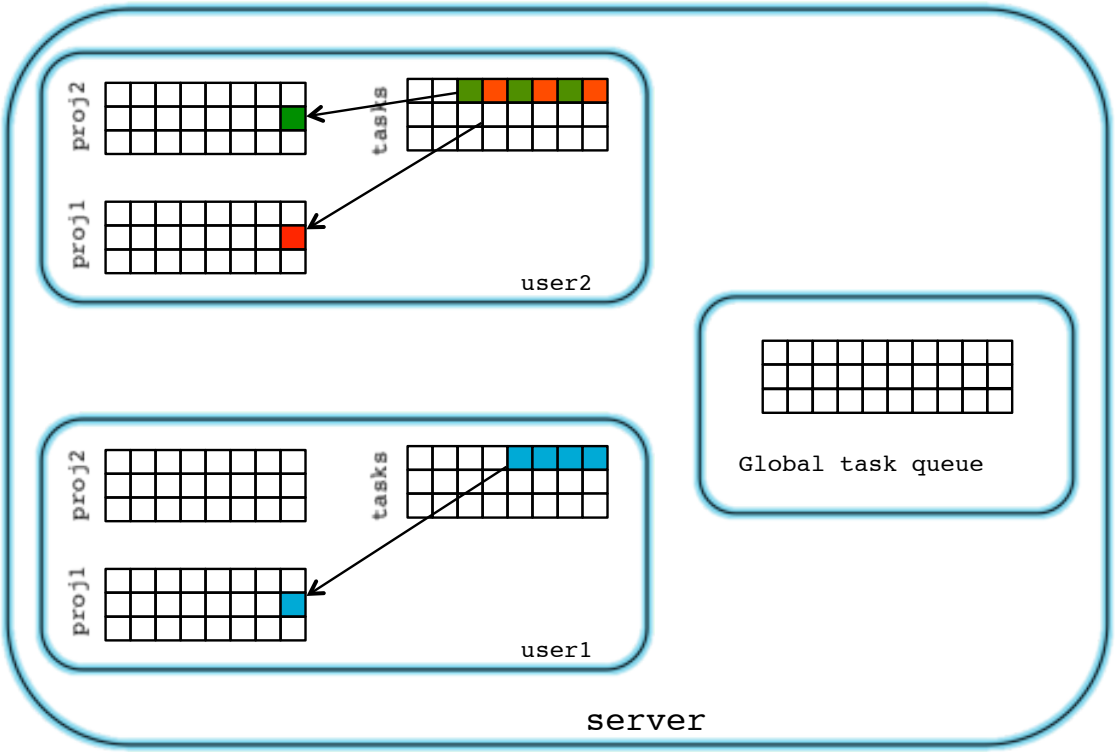


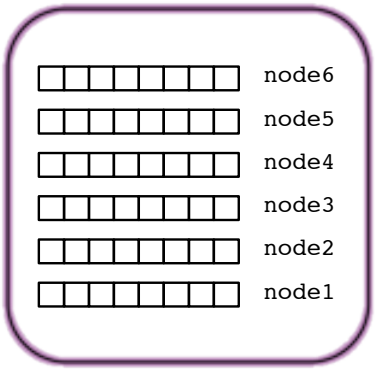
cluster



Completed task files are put into the output task queue of the corresponding user.

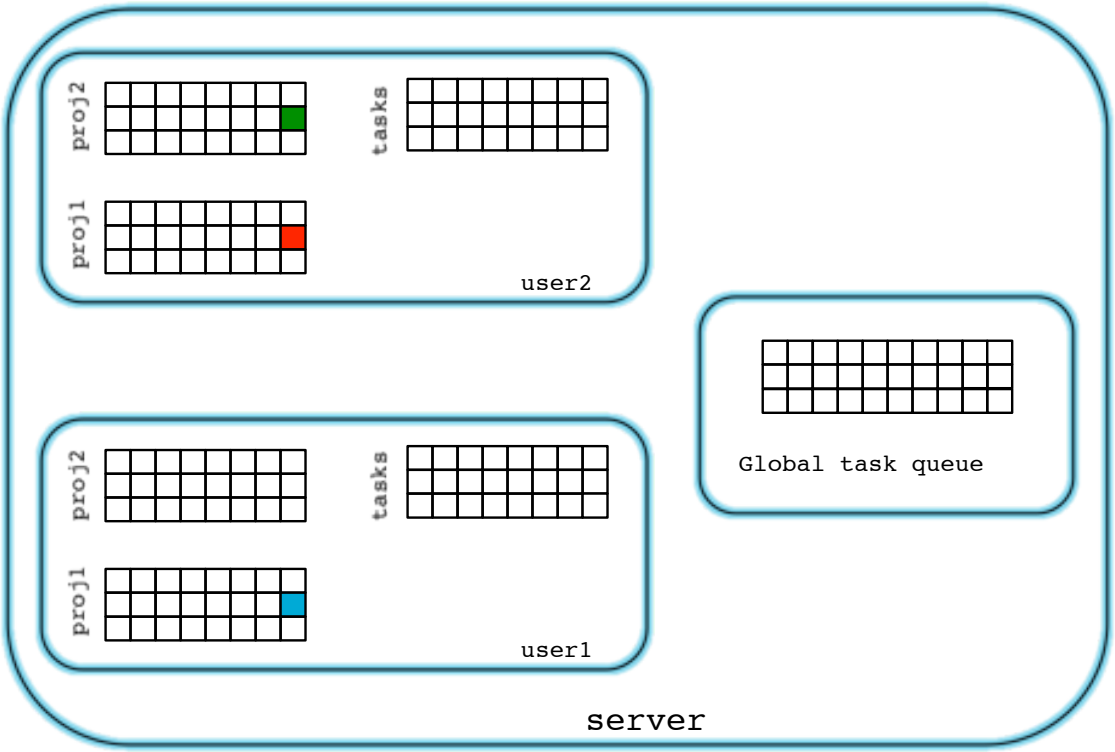
How It Works

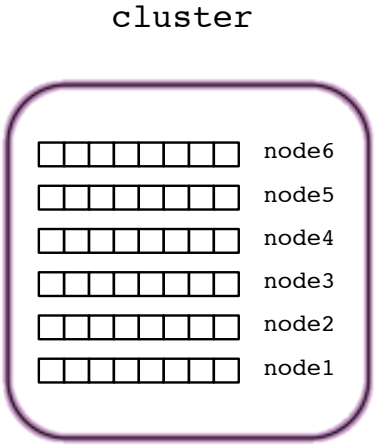




The job manager updates the job file in the running queue,

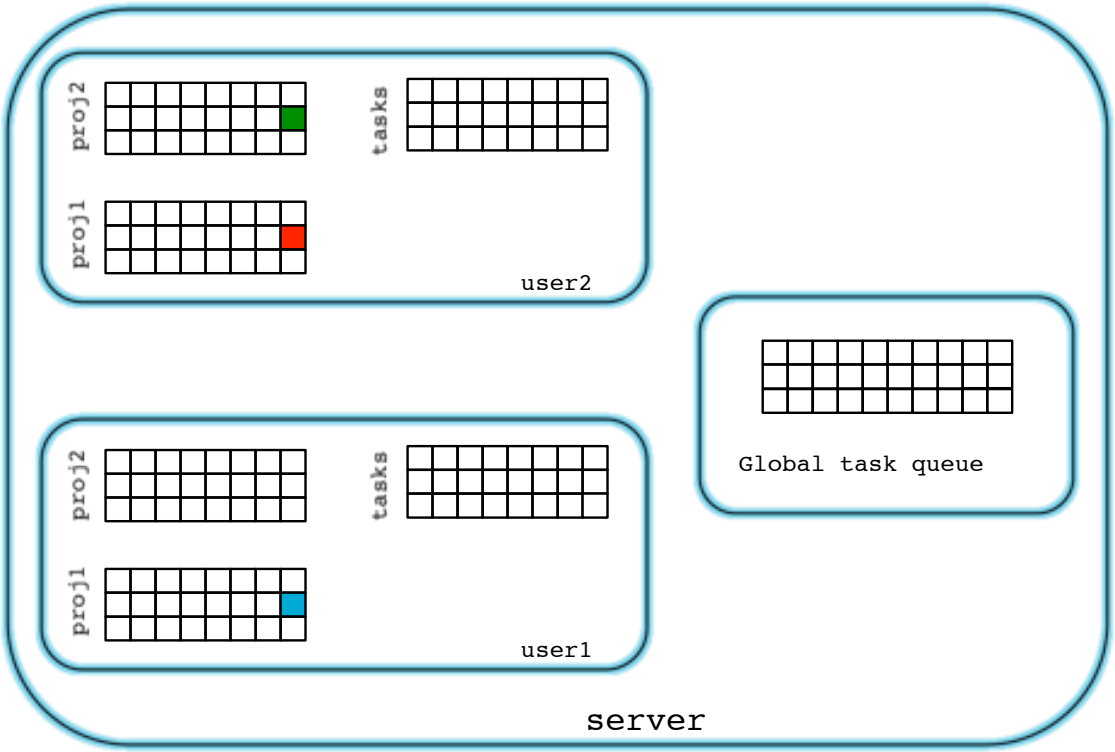
How It Works

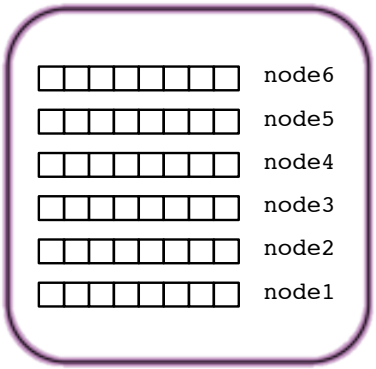




The job manager updates the job file in the running queue, and deletes the output task files.

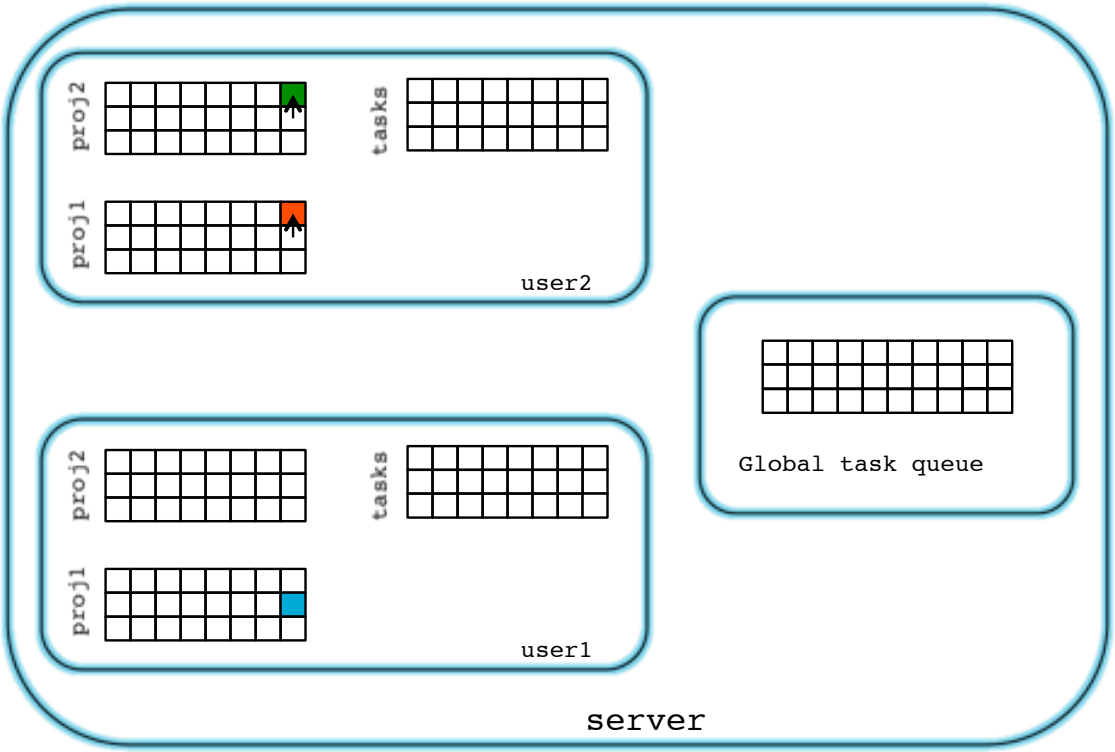
How It Works

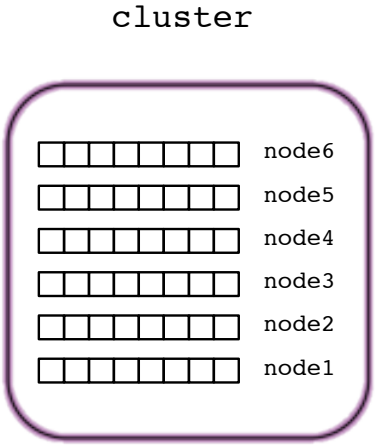




The job manager checks to see if the stopping criteria has been met.

How It Works

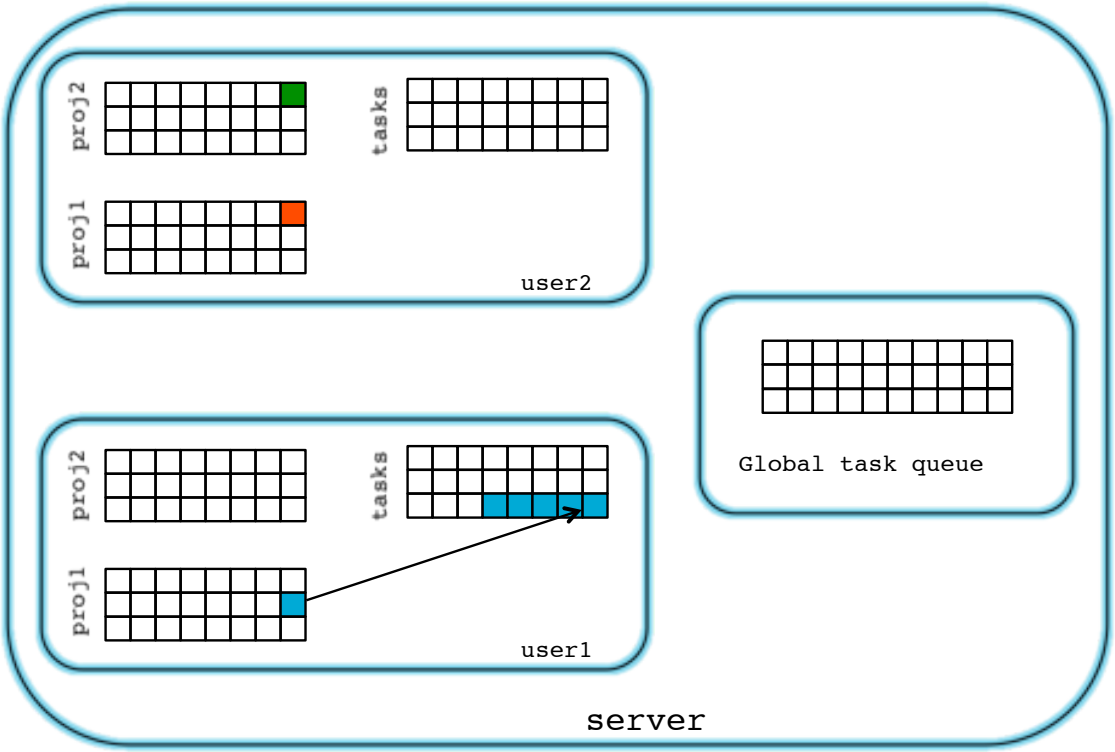


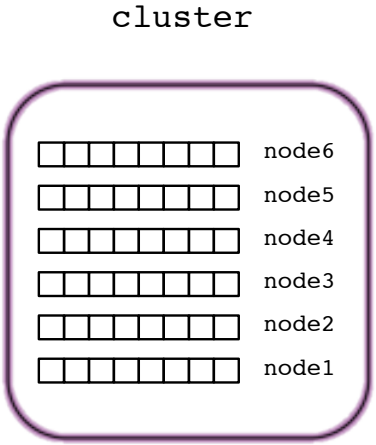


The job manager checks to see if the stopping criteria has been met.

If it has been met, the task file is moved the the output task queue.

How It Works



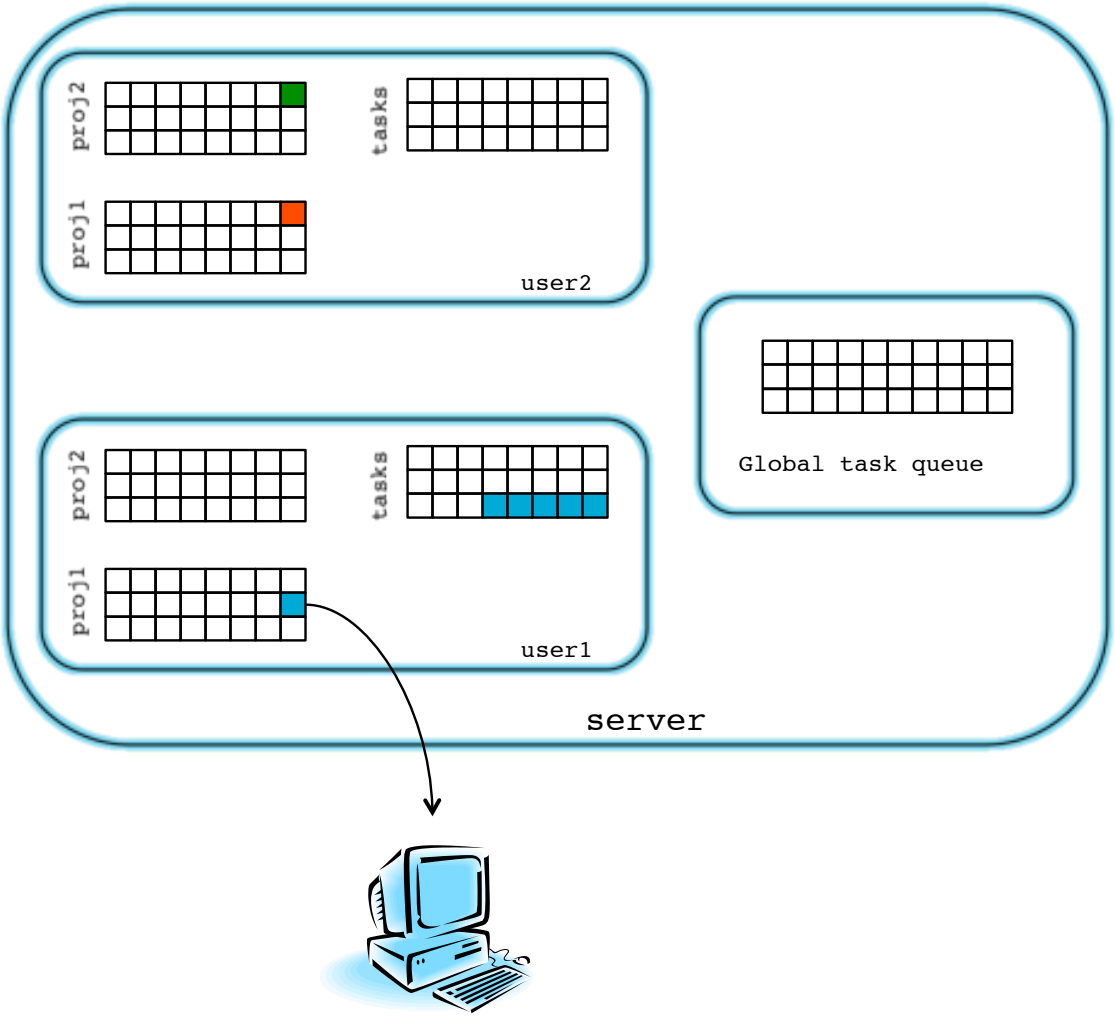


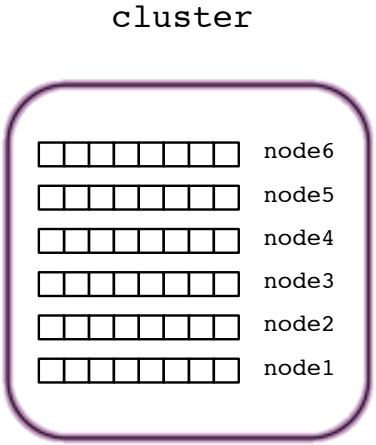
The job manager checks to see if the stopping criteria has been met.

If it has been met, the task file is moved the the output task queue.

If it has not been met, then the job manager creates more task files, and the process repeats...

How It Works





The user can download intermediate or final results through the web interface.

Volunteer Computing

Volunteer computing initiatives create a computing grid using the idle cycles of volunteered computers.

- Examples: Folding@home, SETI@home (Driven by *BOINC* software).

Issues:

1. Should only run when computer is not in use.

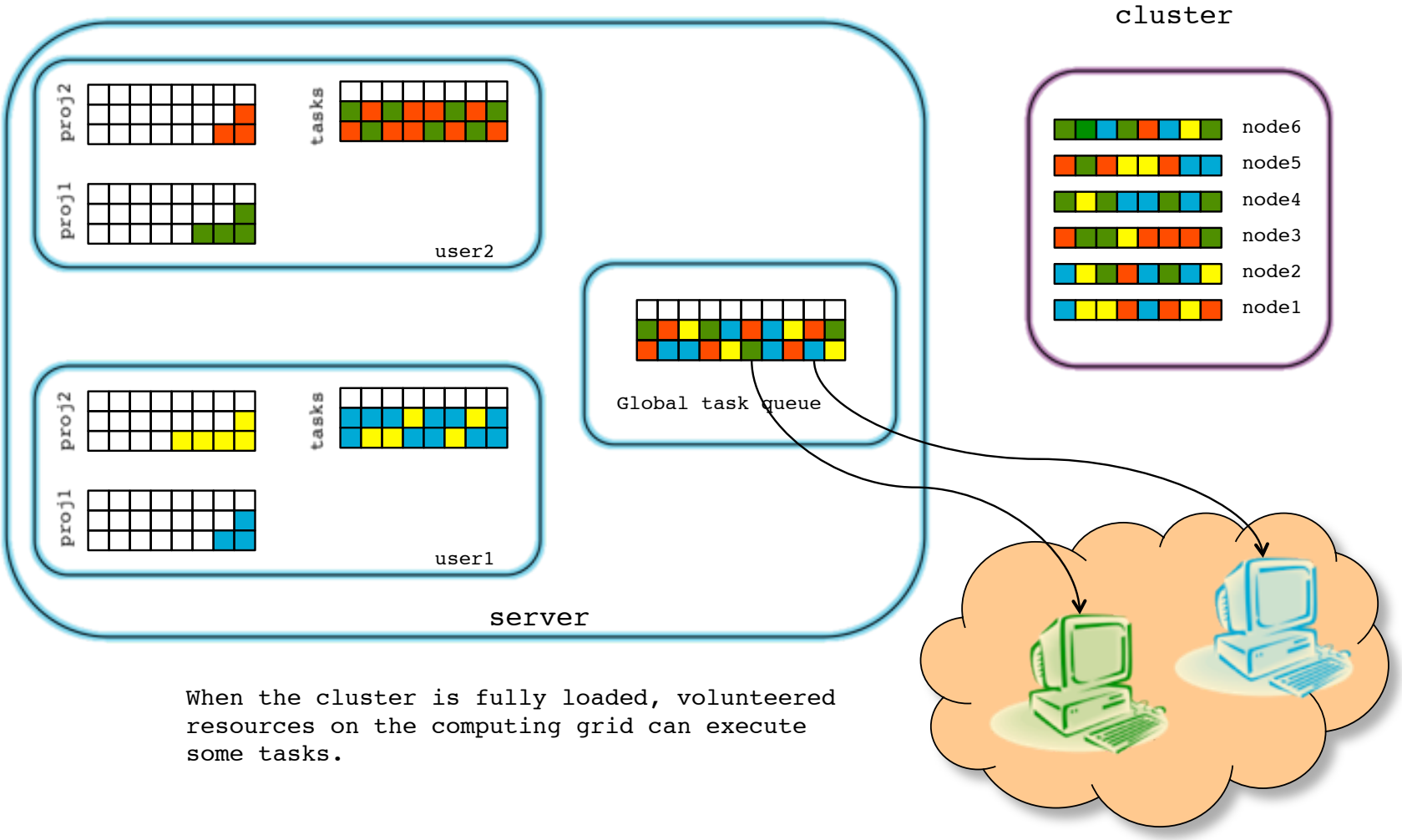
   Solution: Screensaver (windows, mac) or low-priority process (linux).
2. Should work on any common OS.

   Solution: Virtualization.
3. Volunteered computer should not require a copy of Matlab.

   Solution: The Matlab Compiler.

Volunteer computing is facilitated by using **Frontier**, a software solution by Parabon, Inc.

How to Involve Volunteer Computing



When the cluster is fully loaded, volunteered resources on the computing grid can execute some tasks.

Conclusions

- Computing is a commodity.
- By aggregating donated CPU cycles and providing a web-interface, we have developed a computing resource for the communication theory community.
- The computing infrastructure is appropriate for:
  - Parallel Monte Carlo simulation.
  - Exhaustive Optimisation
- We will soon open the resource to the research community.
  - You may apply for an account[2].
- Think about what **you** would do with 100-1000 times more computing power!

[2] http://wcrlcluster.csee.wvu.edu:8180/ldpc/

Thank you