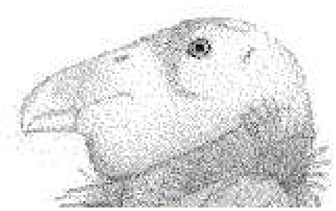
Solving Huge QAPs with Condor

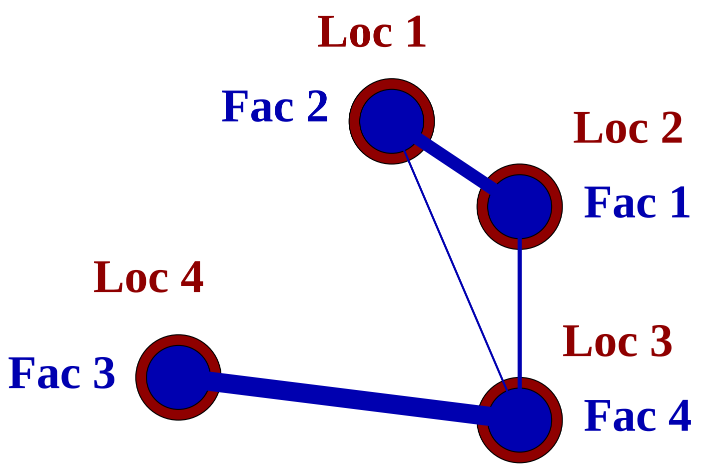
Jeff Linderoth
Jean-Pierre Goux

Kurt Anstreicher
Nate Brixius

Lots of People In This Room



Quadratic Assignment Problem



- Assign facilities to locations
- QAP is NP-Hard
  - No known algorithm is "significantly better" than complete enumeration
  - Examining 10^9 configurations / second, for $n = 27$ would take 345,283,785,211 years, or $>$ 17 Universe Lifetimes.

Miscellaneous QAP Info

Mathematically, the QAP can be stated as

$$\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\pi(i)\pi(j)} + \sum_{i=1}^{n} c_{i\pi(i)}$$

or

$$\min_{X \in \Pi} \operatorname{trace}(AXB + C)X^T,$$

where π is a permutation of $\{1, 2, \ldots, n\}$ and Π is the set of permutation matrices of size $n \times n$.

The QAP has a rather long history, and historically it is one of the “hardest” NP-Complete problems. Practical applications are in facilility layout, the optimal design of typewriter keyboards, telecommunication network design, and others. For references on the QAP, see the QAPLIB page.

`http://www.imm.dtu.dk/~sk/qaplib/`

How Patient are You?

- If 345,283,785,211 years seems a bit long to wait, you might try Branch and Bound.
  - Feasible solution $\Rightarrow$ upper bound
  - Relaxed problem $\Rightarrow$ lower bound

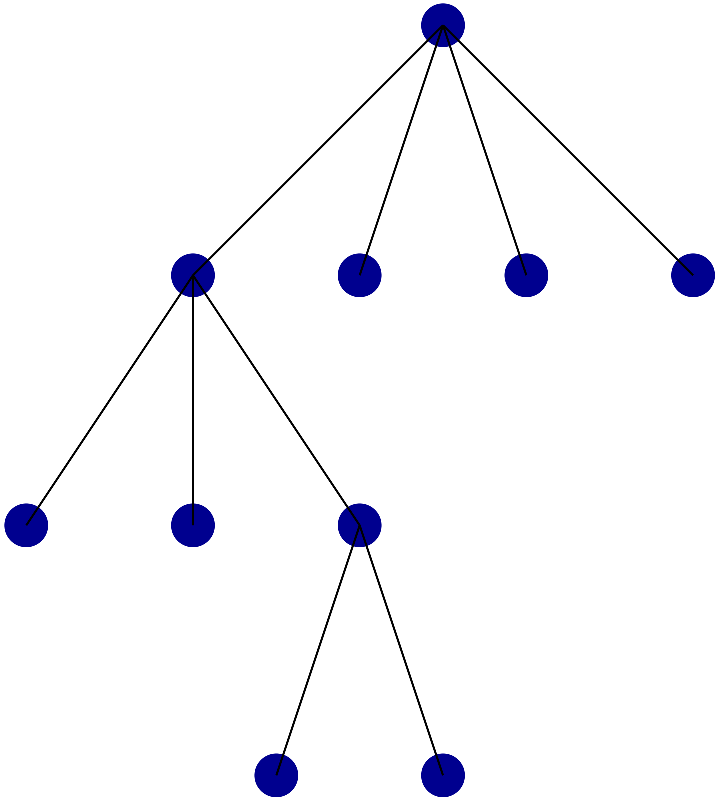
A detailed algorithmic description of branch and bound

1. Is solution to relaxed problem feasible?

Yes? YAHOO!

No? Break problem into smaller pieces. Goto 1.

Branch and Bound



- Conceptually, the there is a search tree than must be explored
- Different nodes are different *Independent* searches

  ⇒ Parallel Computing to the Rescue!

⋆ The size of the tree depends on how good the upper and lower bounds are!

Be Afraid. Be Very Afraid.

The lower bound for the QAP is obtained (in part) by solving the (convex) quadratic optimization problem:

`<Ugly, Ugly Mathematics>`

$$\min \ \mathbf{vec}(X)^T[(B \otimes A) - (I \otimes S) - (T \otimes I)]\mathbf{vec}X + \mathrm{trace}(CX^T)$$
$$\text{such that } Xe = X^Te = e, \qquad X \geq 0.$$

(S and T are obtained from the spectral decompositions of A and B).

`</Ugly, Ugly Mathematics>`

If you don't understand this, join the club! (And I also tried to simplify the exposition!) `:-)`. See [AB00] for the gory details.

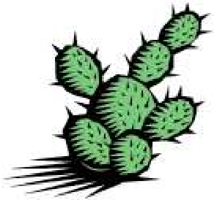
The point is that this quadratic programming problem can be (approximately) solved efficiently, and it produces a lower bound that is "pretty close" to the solution value of the real problem. It is (in order of importance) the combination of the strength of bound, computational efficiency of the bound, and the power of high-throughput computing that make solving big QAPs possible.

MW-QAP – Task definition

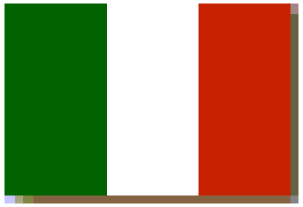
- What should a task be?
  - One node – *Way* too small (milliseconds)
  - One subtree – (Perhaps) too big (Universe Lifetimes)
  - + Compromise – "do subtree for t (300) seconds"
  - Result of task – unfinished nodes and improved solution value
- What does the master do? `act_on_completed_task()`
  - ◇ Puts unfinished tasks in the list
  - ◇ Removes unnecessary tasks if improved solution

Building an Efficient Branch and Bound Solver

- A "Grainsize Analysis" for the QAP showed that...
  - 36% of the tasks have a grainsize smaller that 0.05s !
  - 56% of the tasks have a grainsize smaller that 0.5 !
  - 87% of the tasks $<$ 0.5s rooted at nodes deeper than 6.
- Thus we implemented a "finish up" strategy...
  - Workers work for max_cpu (300) seconds
  - If not finished, allow for extra time to finish up "deep" tasks.
- Short tasks reduced from 56% to 26%. But still too many!
  - For a given level to do the "fast" nodes first
  - Order children in DFS stack based on bound information!
- Hardly any short tasks! Parallel efficiency was up to 95%!

Resources




Number	OS-Arch	Where	How	(Peak) GFLOPS
179	INTEL/LINUX	Wisc	Main Pool	13.88
34	INTEL/LINUX	UNM	Flocked	1.12
64	INTEL/LINUX	INFN	Flocked	2.76
150	INTEL/SOLARIS	Wisc	Main Pool	7.64
35	SUN/SOLARIS	Wisc	Main Pool	1.44
8	SUN/SOLARIS	INFN	Flocked	0.38
32	SGI/IRIX	Argonne	Glide-in	3.84
502	-	-	-	31.06

Jeff's Guide to Flocking and Glide-In

- Flocking

  0. Be nice to Miron – he will get you permission to flock with different pools.
  1. Modify `condor_config` file to set `LOCAL_DIR` and `FLOCK_HOSTS`.
  2. Start your own schedd: `condor_schedd -f -name linderot`

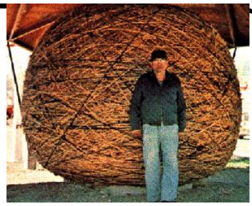
  ⋆ Your jobs will magically be scheduled on machines in the `FLOCK_HOSTS` pools!

- Glide-In

  0. Get account(s) on Globus-enabled remote resources.
  1. Set up Globus identification on local machine. (Copy user certificate and key to known, private location).
  2. `grid-proxy-init`.
  3. `glidein.pl <machine> --count <number>`

  ⋆ `<number>` processors from `<machine>` will magically appear in your Condor pool. If not, contact `jfrey@cs.wisc.edu` and go to 3.

World Records!

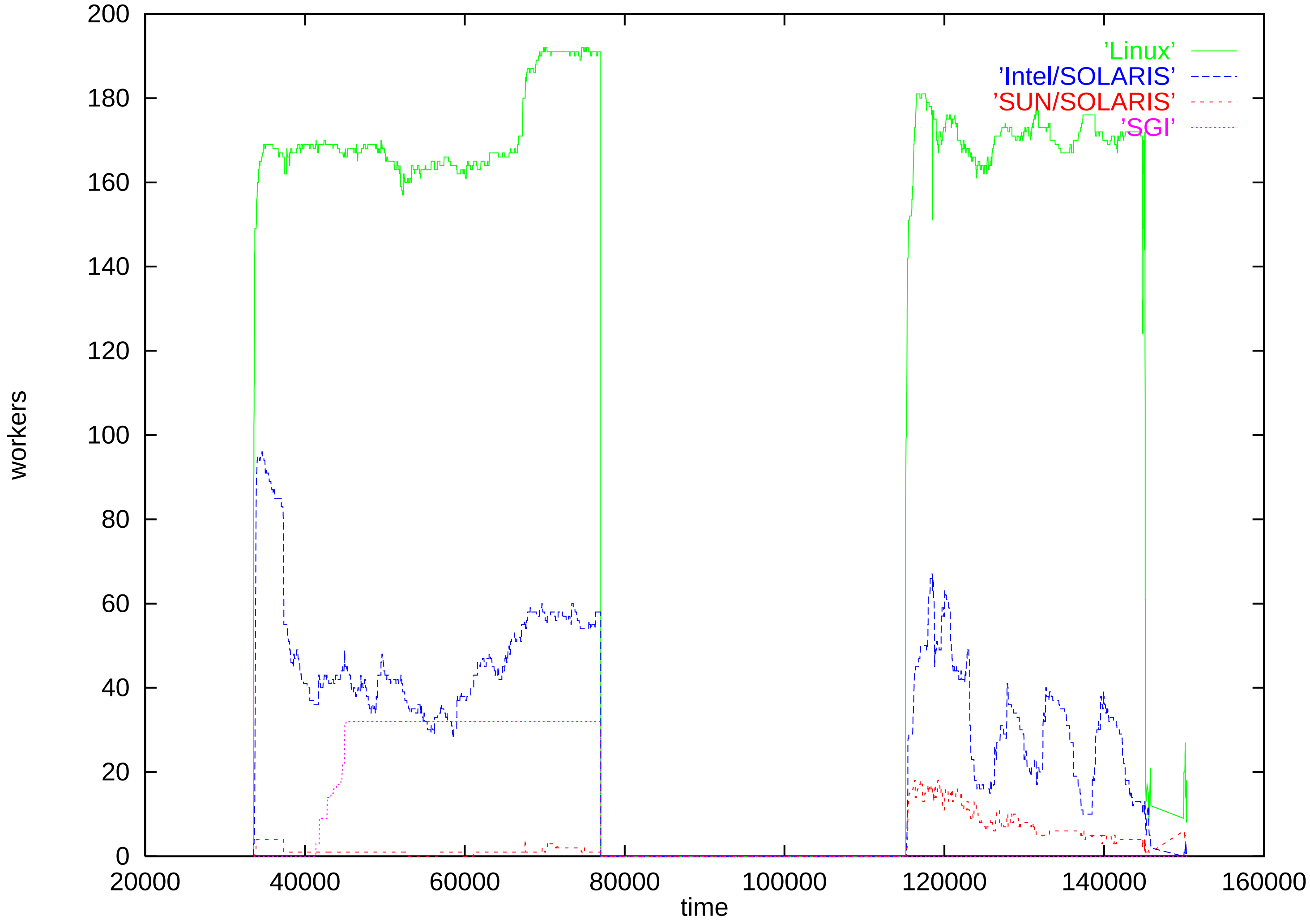


- Previous "record" [HHJ+99] ($n = 25$).
  - Previous time, 66 days (sequential)
  + Our time – 6.7 hours! (Avg. 93 processors, Efficiency $\approx 97\%$.

⋆ New "record" QAP solved [ABG+00] ($n = 27$).

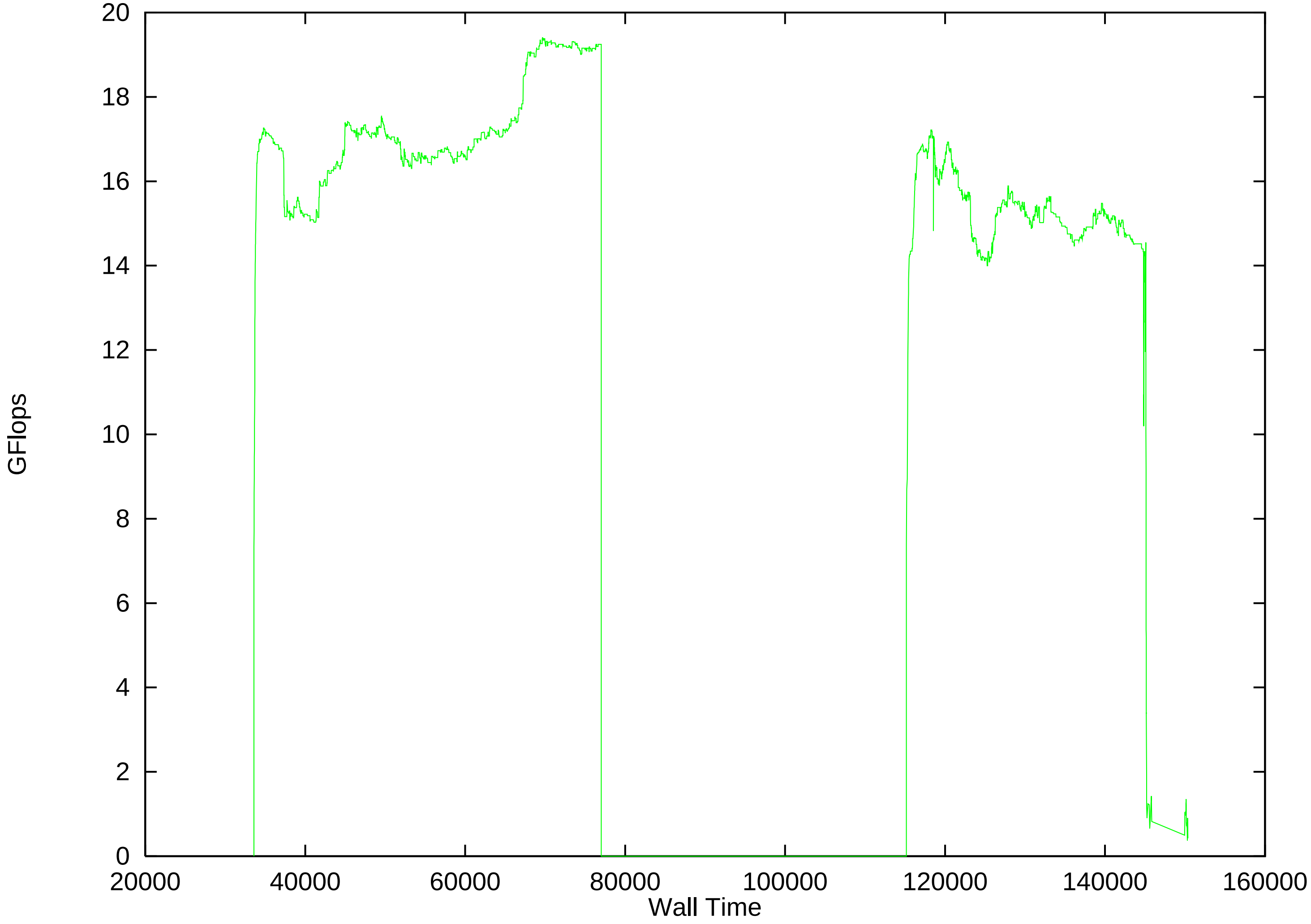
Wall Clock Time:	21:51:17
CPU Time:	152:14:56:40
Nodes:	567,793,866
Avg. Workers:	205
Max Number of Workers:	286
Parallel Efficiency:	$> 81\%$

Workers



'Linux'
'Intel/SOLARIS'
'SUN/SOLARIS'
'SGI'
workers
time
0
20
40
60
80
100
120
140
160
180
200
20000
40000
60000
80000
100000
120000
140000
160000

Performance



That's All Folks!

- Conclusions
  - ◇ MW can do useful work
  - ◇ Condor with Flocking and Glide-in can give you *lots* of computational power
- Future Work
  - ◇ Make MW a little like MS (no need to restart every 12 hours) :-)
  - ◇ Size $n = 28$ and beyond!!!
  - Interactive Problem Solving Environment (iMW). Demo?

References

[AB00] K. Anstreicher and N. Brixius. A new bound for the quadratic assignment problem based on convex quadratic programming. Available from `http://www.biz.uiowa.edu/faculty/anstreicher/qapqp.ps`, 2000.

[ABG+00] K. Anstreicher, N. Brixius, J.-P. Goux, G. Hudek-Davis, and J. Linderoth. Location theory gives rise to QAP problem. *data link*, March 2000. The Alliance Online Technical News Letter, available from `http://www.ncsa.uiuc.edu/SCD/Alliance/datalink/0003/QA.Condor.html`.

[HHJ+99] P. Hahn, W. Hightower, T. Johnson, M. Guignard-Spielberg, and C. Roucairol. Tree elaboration strategies in branch and bound algorithms for solving the quadratic assignment problem. Working Paper, 1999.