NC STATE UNIVERSITY

Review of:

"Decomposition Algorithms for Stochastic Programming on a Computational Grid", Jeff Linderoth and Stephen Wright, 2003

ise

Questions

- What is this paper about?
- Compute the following:

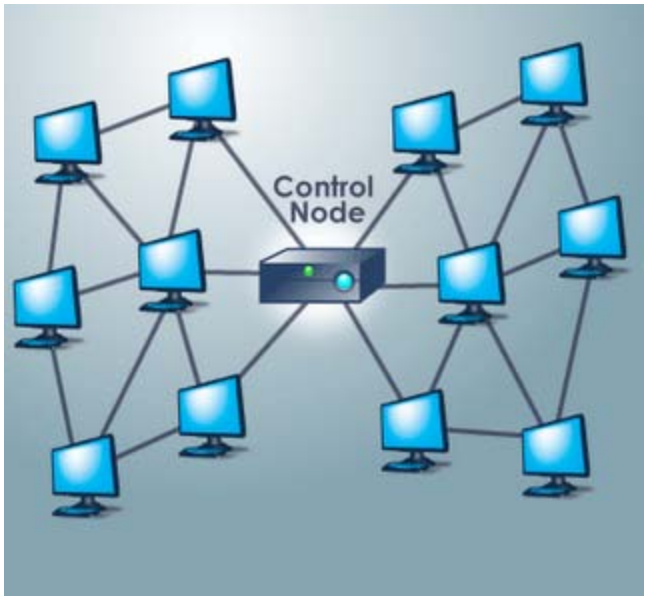
$$Paper \cap Class$$

In other words, what methods described in this paper did we also cover in class?

Grid Computing

- This paper discusses an adaptation of the L-shaped method to leverage parallel computing

- Computational Grid Challenges
  - Geographically distributed
  - Poor communication properties
  - Unreliability of nodes
  - Heterogeneity



More Questions

- Which methods that we have covered in class are most related to those discussed in the paper?
  - Multi-cut L-shaped method
  - Regularized decomposition

- Why the multi-cut L-shaped method? Why not just apply the standard L-shaped method?

Parallel Algorithms

- Parallel implementation of decomposition methods like the L-shaped method takes advantage of separable nature of the recourse function

$$Q(x) = cx + \sum_{i=1}^{N} p_i Q_i(x)$$

- Scenarios can be partitioned into clusters to define discrete tasks for “workers”

Asychronous Algorithms

- In this context an asychronous algorithm considers different points, x, simultaneously (why not synchronous?)
- The master process and second stage (worker) processes can execute in parallel and share information asynchronously
- This is possible since cuts generated at an candidate solution, *x*, are valid everywhere

Questions

- What is the “synchronicity” parameter?
  - Proportion of tasks that must be completed to trigger an update of the master problem

- What is a “trust region” in this paper?

$$-\Delta e \leq x - x^k \leq \Delta e$$

Trust regions radius

Current iterate

Questions

- What is the difference between a “major iterate” and a “minor iterate”?
  - Major iterate $x^k \rightarrow x^{k+1}$
  - Minor iterate $x^{k,\ell} \rightarrow x^{k,\ell+1}$
- What is the “acceptance criteria”?

$$Q(x^{k,\ell}) \leq Q(x^k) - \xi(Q(x^k) - m_{k,\ell}(x^{k,\ell}))$$

Algorithm

ALS: `partial_evaluate`(x^q, q, r)
Given x^q, index q, and task number r, evaluate $\mathcal{Q}_{[j]}(x^q)$ from (7) for all $j \in \mathcal{T}_r$
together with partial subgradients g_j from (9);
Activate `act_on_completed_task` (x^q, q, r) on the master processor.

ALS: `evaluate` (x^q, q)
for $r = 1, 2, \ldots, T$ (possibly concurrently)
`partial_evaluate`(x^q, q, r);
end (for)

ALS: `initialize`
determine number of clusters C and number of tasks T,
and the partitions $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_C$ and $\mathcal{T}_1, \mathcal{T}_2 \ldots, \mathcal{T}_T$;
choose tolerance ϵ_{tol};
choose starting point x^0;
choose threshold $\sigma \in (0, 1]$;
$\mathcal{Q}_{\text{min}} \leftarrow \infty$;
$k \leftarrow 0$, $\texttt{speceval}_0 \leftarrow \texttt{false}$, $t_0 \leftarrow 0$;
`evaluate`$(x^0, 0)$.

Evaluate recourse function

Solve subproblems

Setup L-shaped method

Algorithm

ALS: act_on_completed_task(x^q, q, r)
$t_q \leftarrow t_q + 1$;
for each $j \in \mathcal{T}_r$
    add $\mathcal{Q}_{[j]}(x^q)$ and cut g_j to the model m;
if $t_q = T$
    $\mathcal{Q}_{\min} \leftarrow \min(\mathcal{Q}_{\min}, \mathcal{Q}(x^q))$;

else if $t_q \geq \sigma T$ **and** not speceval$_q$
    speceval$_q \leftarrow$ true;
    $k \leftarrow k + 1$;
    solve current model problem (14) to obtain x^{k+1};
    if $\mathcal{Q}_{\min} - m(x^{k+1}) \leq \epsilon_{\text{tol}}(1 + |\mathcal{Q}_{\min}|)$
        STOP;
    evaluate (x^k, k);
end (if)

Add cuts

Check if sufficient number of tasks completed

Optimality criteria

Trust Region Algorithm

Basic Idea:

- Master node solves the master problem periodically
- Worker nodes simultaneously solve clusters of subproblems and provide optimality cuts when available
- “Acceptance criteria” is used to decide when to update the current iterate, $x^k \rightarrow x^{k+1}$
- Cuts are added and deleted at each minor iteration, $x^{k,\ell} \rightarrow x^{k,\ell+1}$
- The algorithm stops when the current iterate is “close enough” to the optimal solution

Trust Region Algorithm

Procedure Model-Update (k, ℓ)
for each optimality cut
    `possible_delete` $\leftarrow$ `true`;
    if the cut was generated at x^k
        `possible_delete` $\leftarrow$ `false`;
    else if the cut is active at the solution of (17) with positive Lagrange multiplier
        `possible_delete` $\leftarrow$ `false`;
    else if the cut was generated at an earlier minor iteration
        $\bar{\ell} = 0, 1, \ldots, \ell - 1$ such that

$$\mathcal{Q}(x^k) - m_{k,\ell}(x^{k,\ell}) > \eta[\mathcal{Q}(x^k) - m_{k,\bar{\ell}}(x^{k,\bar{\ell}})]$$

        `possible_delete` $\leftarrow$ `false`;
    end (if)
    if `possible_delete`
        possibly delete the cut;
end (for each)

Implementation

- The goal of the Condor® Project is to develop, implement, deploy, and evaluate mechanisms and policies that support High Throughput Computing (HTC) on large collections of distributively owned computing resources…

  http://www.cs.wisc.edu/condor/

  What is “high throughput computing”

Results

- Condor implementation used to solve test instances:
  - CPLEX used for master problem
  - SOPLEX used for subproblems
  - Pool consisted of PCs running linux and PCs and Sun workstations running Solaris
  - High variation in work availibility
  - SMPS format used to define test instances
    - Network design application (SSN)
    - Cargo flight scheduling (Storm)

Results

- Critical parameters:

$$\epsilon_{\text{tol}} = 10^{-5}, \quad \Delta_{\text{hi}} = 10^{3}, \quad \Delta_{0,0} = \Delta_0 = 1, \quad \xi = 10^{-4}.$$

- Cuts deleted if not tight for more than 100 master problems in a row
- Synchronicity parameter varied between 0.5 and 1.0
- Clusters varied from 1 to 14

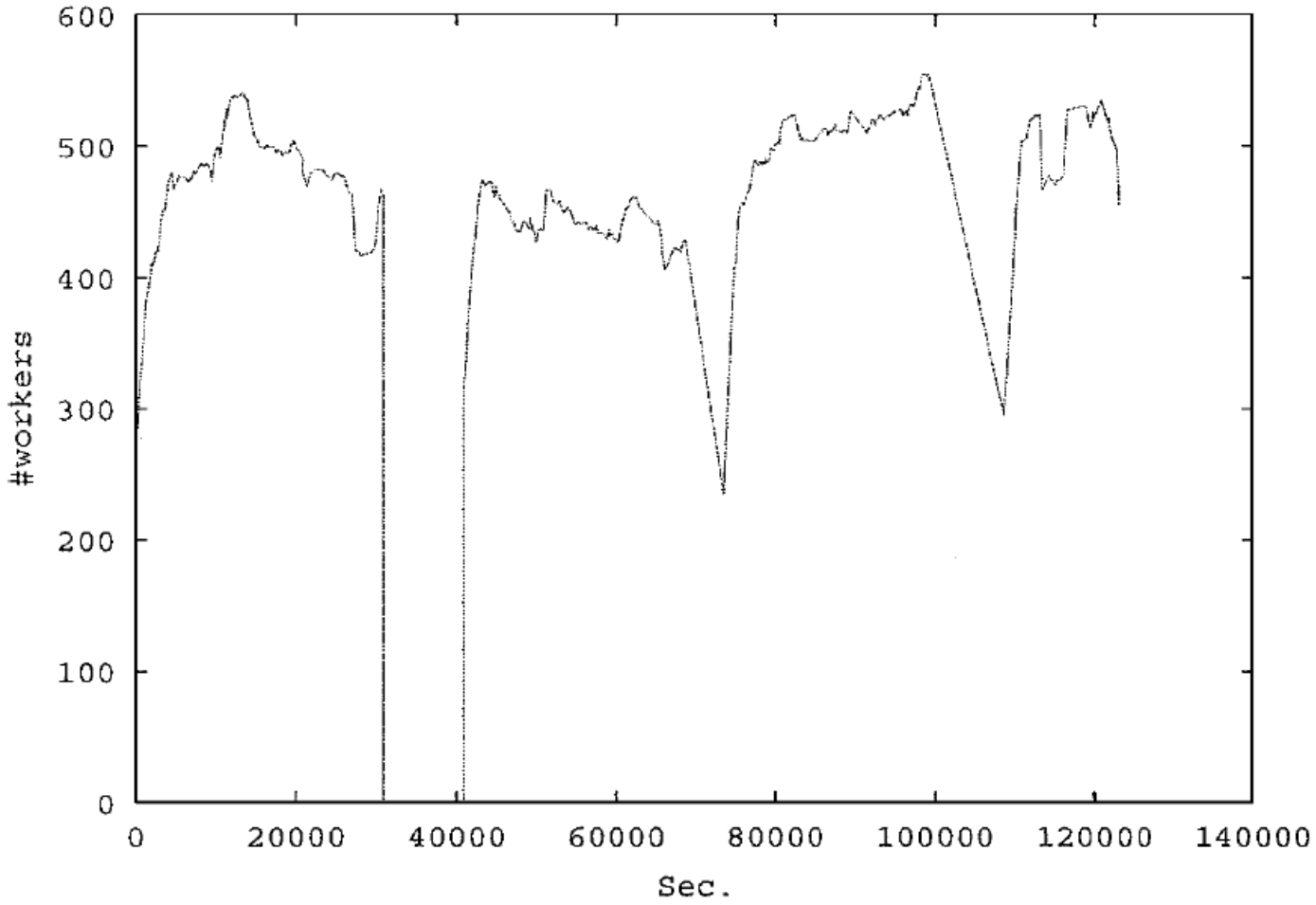
Results

Table 1. SSN, with $N = 10{,}000$ scenarios, Algorithm ALS.

Run	Points evaluated	σ	No. of tasks (T)	No. of clusters (C)	Max. processors allowed	Av. processors	Parallel efficiency	Max. no. of cuts in model	Master problem solve time (min)	Wall clock time (min)
ALS	98	.5	25	200	200	38	.35	19505	8.8	26.2
ALS	93	.7	25	200	200	33	.34	18545	7.9	24.1
ALS	99	.85	25	200	200	34	.25	19776	8.7	33.2
ALS	98	.5	50	200	200	33	.37	19501	8.6	23.6
ALS	97	.7	50	200	200	31	.36	19339	8.6	28.4
ALS	98	.85	50	200	200	32	.33	19573	8.7	29.7
ALS	97	.5	100	200	200	26	.45	19297	8.6	24.8
ALS	106	.7	100	200	200	16	.52	20420	9.6	35.6
ALS	99	.85	100	200	200	29	.41	19771	8.7	22.8
ALS	97	.5	200	200	200	28	.44	19292	8.5	26.2
ALS	99	.7	200	200	200	36	.39	19736	8.9	24.8
ALS	99	.85	200	200	200	40	.32	19767	9.0	27.0

Conclusion: not very sensitive to σ or T

Results


600
500
400
300
200
100
0
#workers
0
20000
40000
60000
80000
100000
120000
140000
Sec.

Results

- Comparison to regularized decomposition:

Table 8. Storm, with $N = 10^7$ scenarios.

Run	Points evaluated	$\|\mathcal{B}\|$ (K)	No. of tasks (T)	No. of clusters (C)	Max. processors allowed	Av. processors	Parallel efficiency	Max. no. of cuts in model	Master problem solve time (hr)	Wall clock time (hr)
ATR	38	4	1024	1024	800	433	.668	39647	1.9	31.9

Table 10. RD code performance on problem storm.

N	Starting point	Iterations	Final objective	Time (min)
1,000	Cold start	38	15508008.1	13.1
10,000	Warm start ($N = 1{,}000$)	41	15499432.7	255.6
100,000	Warm start ($N = 10{,}000$)	3*	–	877.0*

*Terminated prior to convergence.

Conclusions

- This paper presented a trust-region approach suitable for asynchronous parallelization
- Proof of convergence
- Design and implementation of the methods to solve large sampled tests instances
- This paper demonstrates that it is possible to solve large stochastic programs efficiently with inexpensive computational platforms

ISE789 Wrap-up

Course Objectives:

- To build a general understanding of Stochastic Programming and the theory and methodologies identified with it
- To understand state-of-the-art in the research literature
- To set a foundation for future research in stochastic optimization

Next Time

- Project Presentations by Alex, Behzad, Bjorn, Brian, Clay, Hamed, and Nils
- Max of 10 minutes
- Send me your slides by noon on Tuesday