Beyond worst-case analysis:
Observed low depth for a P-complete problem

Uzi Vishkin
Design and analysis of algorithms: core theory topic to CS majors

• Nearly all examples in popular textbooks involve:
  – Design for worst case performance, and
  – Analyze for the worst case

Take home message from “theory”

Worst-case performance is the silver bullet

In particular: NP-Complete ➔

Problem (as-is) is
Surprise 1: Practice is... “misbehaving”

• Some of the most commercially impactful algorithms do not fit the worst case mold

• **Examples** Algorithms for:
  – Clustering
  – Linear programming
  – Neural network training
  – SAT solvers (quintessential NP-complete problem)

• **Metric:** *wall-clock time*

• **Beyond Worst-Case (BWC):** NP-complete?... let’s see...


**Upshot** for “theory”:

**Worst-case?** ...You risk missing big impact opportunities
Surprise 2: No parallel algorithms in edited book

• Since 2005, parallelism leads performance growth opportunities
• Worst-case is mostly about performance, and so is ...BWC...
• Still: no parallel algorithms in the edited book
  – Note: parallel matrix multiplication - a heavily used routine. BUT in a worst-case form. Only the (serial) calls to the routine are where BWC tends to happen
• Let’s ponder: how come? how could BWC application emerge? Wall clock trial&error by parallel algorithmicists and programmers

Surprise (?) 3: No commercial “general-purpose” manycores. Too few program or taught to program in parallel.
[Beyond current scope: Vendors’ fault or theory fault?]
In any case, where could empirical evidence emerge from?
Back to complexity theory

• NP-complete $\Rightarrow$ “no” polynomial time serial algorithm. Still, practical BWC algorithms for some NP-complete problems

• P-Complete $\Rightarrow$ ”no” parallel algorithm with poly-log worst-case time & poly processor#

“nightmare for parallel processing”

[MehlhornSanders08]

• BWC: ?

Surprise 4: A first very efficient BWC parallel algorithm for a P-Complete problem—The current paper

• Wait, didn’t you just tell us that BWC correlates with wall clock time, but the manycore needed is not available?

• Generally, a chicken-and-egg impasse, but the paper circumvents it by resorting to a model. Hence: PMAM.
3 CNF Horn-SAT

Example: C1 = (x1 V ¬x2 V ¬x3), C2 = (x2 V ¬x3 V ¬x4), C3 = (x3 V ¬x1), C4 = (x1)

Per clause: ≤3 literals. ≤1 positive.
Formula: C1 ∧ C2 ∧ C3 ∧ C4
Problem: is the formula satisfiable?
Known: 3 CNF Horn-SAT: (i) has linear time serial algorithm, but (ii) is P-Complete
Loop of the parallel algorithm: “parallel unit propagation”

While there are clauses with singleton literals (“unit clauses”)
  Satisfy all of these literals
In parallel, “Remove”:
  (i) all satisfied clauses, and
  (ii) all implied FALSE literals in every clause in which they appear.
For every affected clause, perform the following case analysis:
  a. No literals remain. Conclude: the formula is unsatisfiable. Terminate the algorithm.
  b. Only one literal remained. Include this literal in the set of singleton literals for the next iteration

Final step. If the above did not conclude that the formula is satisfiable or unsatisfiable, conclude: The formula is satisfiable. Assign FALSE to all remaining variables to derive a satisfying assignment.

N – total number of literals.
h – number of iterations. Namely: length of the longest chain of such commit iterations, given the input formula (by limited analogy to BFS).

On a PRAM: \( O(N/p + h \log N) \) time or \( O(N/p) \) time for \( p \leq N/(h \log n) \)
Recall: Circumvent chicken\& egg? \( h \) dominates. So, just observe values of \( h \).
Random Horn SAT formulae model [Moore 2007]

Given $d_1 < 1$ and $d_3$, the random Horn-SAT formula $H(N,d_1,d_3)$ is the conjunction of:

1. a single negative literal $x_1$;
2. $d_1N$ positive literals chosen uniformly from variables $x_2, \ldots, x_N$;
3. $d_3N$ 3-literal clauses. 3 variables are chosen uniformly. One positive literal.

Experiments 3 values for $N$: 10,000, 100,000, 1,000,000. $0.2 \leq d_1 \leq 0.2$, $1 \leq d_3 \leq 5$.

Figure shows: average value of $h$ exceeded 60 only for one category: $d_3=2$ and $d_1=0.18$.

Average value of $h$ for each category (number of variables, $d_1$ and $d_3$)
Largest observed value of h

Figure shows: largest value observed was slightly above 200 for the only category where the average value of h exceeded 60. Note: for N=1,000,000, h is lower than N by nearly 4 order of magnitude or more \(\Rightarrow\) low parallel time. Noteworthy for our P-complete problem.
Horn SAT formulae derived from an application

• SAT formulae models a question (e.g., correctness, reachability) reduced to satisfiability

• Results are even more appealing for h for such an application

• For instances that led to over 100M variables the largest h observed was only 18. Hardly a nightmare...
Conclusion

• The current paper used an ad-hoc model for: theory nightmare ➔ wake up to rosy BWC outlook. But, essentially it is a PRAM algorithm
• Prior UMD “PRAM-On-Chip” HW-SW prototyping
• IMO, HW vendors bungled up so far the general-purpose manycore opportunity
• Competition among CPU vendors is becoming fierce again for the first time since the 1990s
• How to get to general-purpose manycore is beyond scope for current paper