EMBRACING UNCERTAINTY: PROBABILISTIC PARALLELISM

ANDRÁS VAJDA, ERICSSON

JOINT WORK WITH PER STENSTRÖM (CHALMERS)

(AND BASED ON THE WORK OF MANY OTHERS ACROSS OUR FIELD)

ABOUT ERICSSON

- World's largest telecom equipment provider
  - •40% overall market share, 50% in 3G
- World's 5th largest, Europe's 2nd largest software company
  - based on sales values

GLOBAL SOFTWARE TOP 100 - EDITION 2010

29 September 2010 , Written by Michel van Kooten

SHARE

This list contains the top companies in the software industry, ranked by annual software revenues. Software revenues are defined as revenues coming from sale of licenses, maintenance, subscription and support. Revenues from custom software development are excluded.

Please read the research methodology if you wish to gain a deeper understanding of the way the list was composed.
More information on the world's largest software companies can be found in this article with software industry analysis, trends and highlights.
The previous (2009) edition of the Global Software Top 100 is available here.

#	Company	Software Revenues *mln US$*	Software Revenue growth	Total Revenues *mln US$*	Software Revenue share
1	Microsoft >	49,090	-1%	61,159	80%
2	IBM >	21,396	-3%	95,758	22%
3	Oracle >	18,582	6%	22,734	82%
4	SAP >	11,368	2%	15,373	74%
5	Ericsson >	7,595	5%	29,014	26%
6	Nintendo >	6,799	-6%	17,762	38%
7	HP >	6,183	-15%	116,245	5%
8	Symantec >	5,565	-2%	5,992	93%
9	Nokia Siemens Networks >	4,529	-15%	18,114	25%
10	Activision Blizzard >	4,279	-7%	4,279	100%
11	CA >	4,013	-3%	4,219	93%

OUR REAL DILEMMA…

- Moore’s law is alive and kicking …

- … but we don’t have a reliable method to use the transistors for running applications with strong sequential dependencies faster

We are stuck.

ONE VIEW FROM A SOFTWARE GUY....

› Computer architecture failed to address this dilemma

› Many reasons
- … too strong insistence for too long on the ISA as THE INTERFACE
- … hence, too little knowledge of what the program is really up to
- … focus too much on executing the **how** instead of focusing on **what** to execute
- … while the software guys ignored execution

ANOTHER VIEW FROM A SOFTWARE GUY…

- The Newton era of physics
  - Everything is deterministic, governed by strict laws
  - Time is absolute
  - Position of everything can be precisely calculated
- The quantum era of physics
  - Everything is probabilistic and relative
  - Particles can take infinite forms and places
- The von Neumann era of computing
  - The code to execute is pre-determined
  - Dependencies are absolute and execution order cannot be changed

OBSERVATIONS...

- Available *possible parallelism* depends on the data, not just the algorithm
- Dependencies are not created equal: some are weaker than others
  - … only hold if some conditions are met
  - … the source of the dependence can only take a few finite values
- There may be many paths to execute a program without dependencies
  - … of which one is the observable (and wanted) path

PROBABILISTIC PARALLELISM

- Parallelism that ***may be available***, characterized by….
  - … *probability*: the probability that the parallelism will actually be achieved
  - … *uncertainty range*: set of alternative execution possibilities of which all have to be explored to reach the probability level

- Turning probabilistic parallelism into true parallelism usually depends on the *nature of the input data*
  - … different from traditional sources of parallelism: algorithms

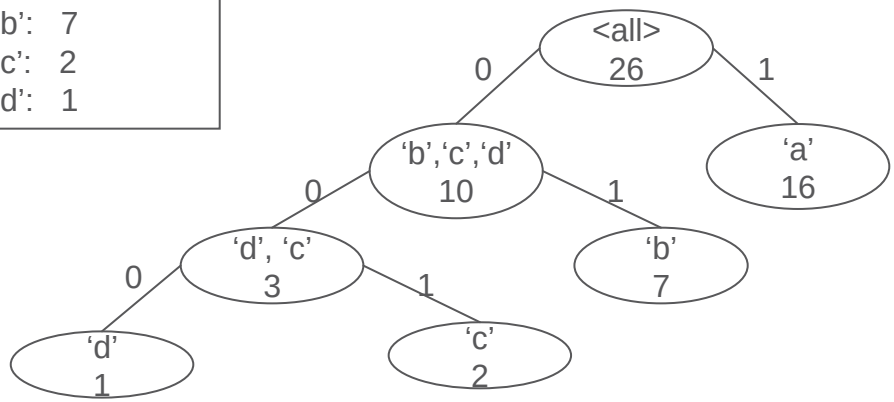
PROBABILISTIC DATA PARALLELISM (1)

- Motivating example: Huffman decompression
- Basic algorithm: assign shorter codes to more frequent symbols
- Decompression: hard to parallelize, as it's hard to deterministically identify where code boundaries are
- Idea: it's possible to calculate the probability ***P*** of having a code boundary within ***N*** consecutive bits
- Solution: apply data parallelism, but for each chunk, execute ***N*** parallel *speculative* version, each starting at different bit positions

Original String: aaaabbbcaaababababaaaadcaaab

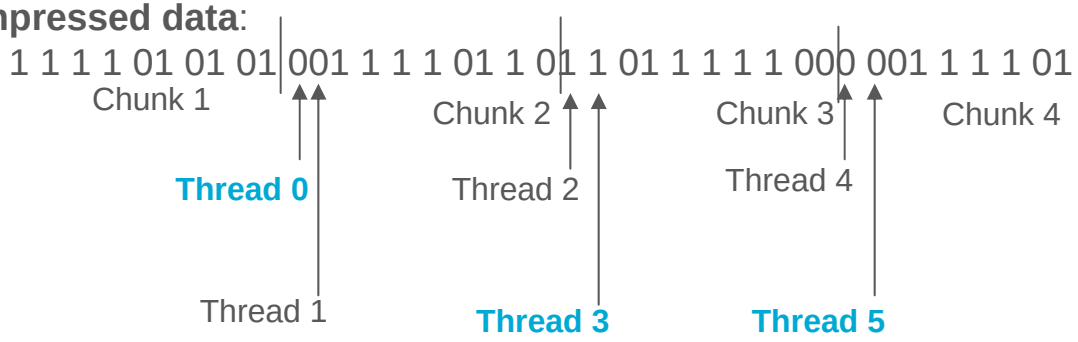
Frequencies:
'a': 16
'b': 7
'c': 2
'd': 1

Code Tree:



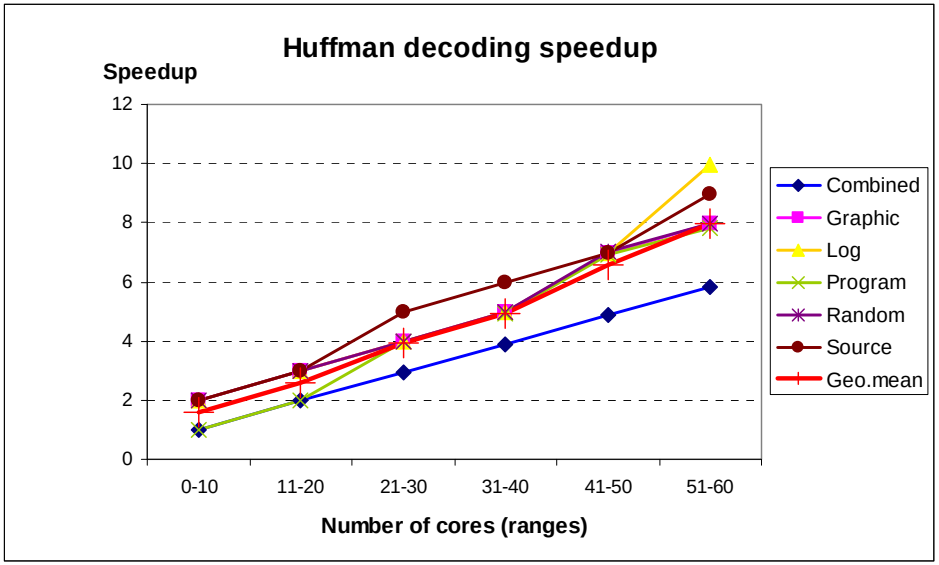
Depth level: 2 Probability: 23/26 (89%)

Compressed data:



Successful threads: 0, 3, 5

PROBABILISTIC DATA PARALLELISM (2)



- In Huffman decompression case:
  - Spatial Interval: ***N*** (in the example, 2)
  - Associated probability: ***P*** (in the example, 0.89)
  - Speedup on K cores for the example is

    S= (K-1) / 2 + 1,

    with probability 0.89

- Probabilistic data parallelism: data parallelism where chunk boundaries are defined as a ***spatial interval*** with ***associated probability***

- Probable speedup on K cores, with probability P:

  S = (K-1) / N + 1

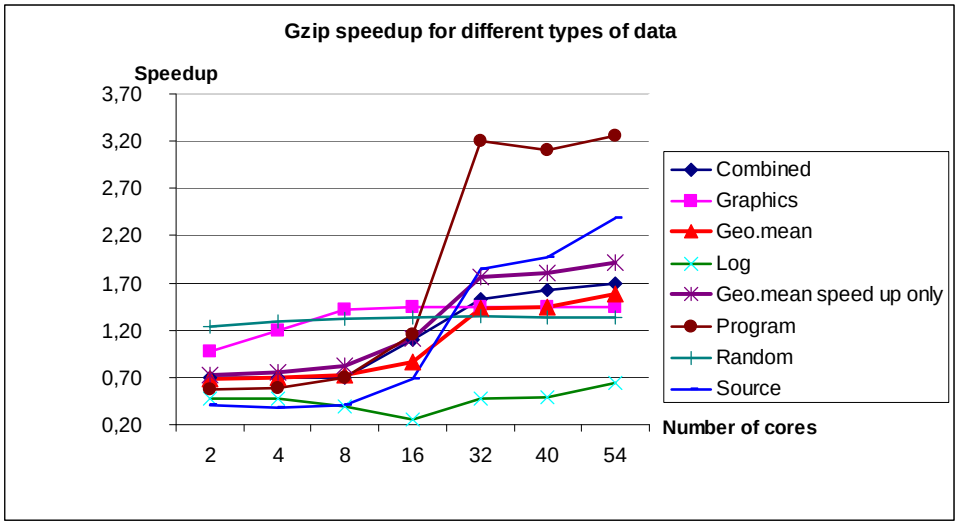
- Speedup is dependent on the number of cores, the algorithm and the input data

PROBABILISTIC TASK-BASED PARALLELISM

- Motivating example: 164.gzip
- Hard to parallelize algorithm…
- … however some parts can be pre-computed without significant impact on semantic outcome

- Probabilistic task-based parallelism: pre-execution of parts of the code (tasks), with the result made available if and when needed

```
deflate () {
  while (lookahead != 0) {
    INSERT_STRING();  match = longest_match();
    if (match) {
      reset_needed = ct_tally_match(match);
       while (matchProcessed) INSERT_STRING(); }
    else reset_needed = ct_tally_match(currentSym);
     if (reset_needed) FLUSH_BLOCK();
     while (lookahead < MIN_LOOKAHEAD)
  fill_window();
 } }
```



EXPRESSING PROBABILISTIC PARALLELISM

- Semantic hints: programmer indications to guide the run-time system
  - No change to original code: ignoring the hints results in sequential execution
- We may need new language and ISA constructs
  - Open research

EXAMPLE: PROBABILISTIC DATA PARALLELISM HINTS

- Shall express …
  - … which part of the application exposes probabilistic parallelism
  - … what is the uncertainty range (interval for data parallelism boundaries)
  - … how to choose the correct result of the execution

- Shall result ….
  - … in multiple executed variants
    - … of which one (or none) is chosen as the correct one
  - … in case of failure, deterministic re-execution of the code

ARCHITECTURAL MECHANISMS

- Isolated execution of bits of the code
  - Similar to speculative execution, but at a coarser level
- Private memory versions
  - Can exploit HW/SW transactional memory mechanisms
- HW is not enough: SW support is needed
  - Orchestration of probabilistic executions
  - Winner selection

THANK YOU!

ANDRÁS VAJDA

ANDRAS.VAJDA@ERICSSON.COM

BLOG: WWW.A-VAJDA.EU/BLOG

TWITTER: @andrasVajda

ERICSSON