A Scalability Metric for Parallel Computations on Large, Growing Datasets (like the Web)

Jesse Weaver (@jrweave)
Tetherless World Constellation
Rensselaer Polytechnic Institute

November 11, 2012
Outline

1. Introduction
2. The lack of a data-centric scaling perspective
3. Proposing data scaling and growth efficiency
4. Caveats, benefits, and an example
5. Conclusion
Introduction

- Challenge: scaling to quantities of data found on the Web.

- Certainly an issue for reasoning over RDF data.

- Parallelism is a common solution to handling more data.

- However, traditional parallel computing metrics seem unfit for measuring this particular scalability challenge.
Motivating Assumptions

- Important to scale toward the size of the Web.
  - State of the art: ≈1 billion, real-world RDF triples\(^1\).
  - The Semantic Web consists of over 24.7 billion RDF triples [Biz10].

- The Web is continuously growing.
  - Even scaling to 24.7 billion triples is not enough.

- Thus, we need to somehow measure or demonstrate the ability of these systems to scale to larger datasets.

\(^1\)[HPPD10, KOvH10, UKM\(_+\)10, WWAH10]
Preliminaries

Definition: A **growing dataset** is a function $D$ that maps positive integers to datasets such that for any positive integer $n$, $|D(n)| = n$ and $D(n) \subseteq D(n + 1)$.

Notation: $T_D(P, k)$ is a function denoting the time for $P$ processors to execute with input $D(k)$.

Question: How does parallel execution time $T_D(P, k)$ change when number of processors $P$ increases to accommodate larger datasets ($D(k)$ as $k \to \infty$)?
The lack of a data-centric scaling perspective

We have defaulted to strong scaling.

- Fix the problem size by fixing $k$.
- Increase processors $P$ to decrease execution time $T_D(P, k)$.

$$S(P) = \frac{T_D(1, k)}{T_D(P, k)}$$

- Inapplicable to our particular challenge because $k$ does not increase with $P$.
- Unfortunately, metrics based on speedup are the most commonly used\(^2\) (others used only variants of execution time\(^3\)).

\(^2\)[GM10, KOvH10, OKA\(^+\)09, SP08, UKM\(^+\)10, UKOvH09]

\(^3\)[HPPD10, KMK08, WH09]
Fix a parameter *other than problem size*.

- Increase processors $P$ to accommodate larger problem size.

But how is “problem size” defined?
- Usually sequential execution time (or “workload”).
- **For our case, should be (input) dataset size.**
Fixed-time scaling [Gus88] does not fit.

- Fix a function $f$ such that $\forall P$, $T_D(1, f(1)) = T_D(P, f(P))$.

- Increase processors $P$ to increase workload $T_D(1, f(P))$.

  $$S_{\text{scaled}}(P) = \frac{T_D(1, f(P))}{T_D(1, f(1))}$$

- At least now the dataset changes.

- **However**, $f(P)$ may decrease as $P$ increases.
Memory-bounded scaling [SN93] is similar.

- Let $g$ be a function such that for $P$ processors, $g(P)$ is the largest size such that no processor exceeds its memory usage when computing for input $D(g(P))$.  

- Increase processors $P$ to increase overall memory.

$$S_{MB} = \frac{T_D(1, g(P))}{T_D(1, g(1))}$$

- At least now it seems that $g(P)$ would increase with $P$.

- However, $g(P)$ is difficult to know empirically for some problems, and memory-bounded speedup can be hard to measure.
Fix $k$ and call it the **processor capacity**.

Increase processors $P$ to accommodate data $P \cdot k$.

The growth efficiency is given by:

$$\text{growth efficiency} = \frac{T_D(1, k)}{T_D(P, P \cdot k)}$$

Now the relationship between dataset size $P \cdot k$ and processors $P$ is obvious (linear).
Caveats of Data Scaling

- Must justify notion of **processor capacity**.

- Must clearly define the **growing dataset** used in evaluations.

- As with relative speedup, growth efficiency is meaningful for comparing systems *only* when accompanied by execution time.
Benefits of Growth Efficiency

- A direct measure of data scaling.

- Amenable for comparing systems when:
  - reported along with execution times;
  - using sufficiently similar notions of processor capacity;
  - using sufficiently similar, growing datasets; and
  - plotted over dataset size instead of number of processors.

- Unlimited empirical measure of weak scaling.
  - Unlike with common weak scaling metrics, no need to time larger workloads on only one processor.
    (i.e., no $T_D(1, f(P))$ as $P \to \infty$)
Retrospective [WH09]

RDFS materialization; LUBM dataset; processor capacity of 2,699,360 triples; up to 256 processors.

Fig. 1: Efficiency and growth efficiency (log/log) up to 256 processors
Conclusion

- Need to move focus of evaluations toward data scaling, or at least weak scaling in general.

- Existing weak scaling metrics are limited for empirical evaluation because they are based on sequential execution time (or “workload”).

- Growth efficiency is a data scaling metric, and it is useful for empirical evaluation with any number of processors.
Questions?

(Quick... raise your hand if you’re thinking about isoefficiency.)
Christian Bizer.
Pay-as-you-go Data Integration on the public Web of Linked Data.
Keynote Presentation at the 3rd Future Internet Symposium, September 2010.

Eric L. Goodman and David Mizell.
Scalable In-memory RDFS Closure on Billions of Triples.

John L. Gustafson.
Reevaluating Amdahl’s Law.

Aidan Hogan, Jeff Z. Pan, Axel Polleres, and Stefan Decker.
SAOR: Template Rule Optimisations for Distributed Reasoning over 1 Billion Linked Data Triples.

Zoi Kaoudi, Iris Miliaraki, and Manolis Koubarakis.
RDFS Reasoning and Query Answering on Top of DHTs.
Spyros Kotoulas, Eyal Oren, and Frank van Harmelen.  
Mind the Data Skew: Distributed Inferencing by Speeddating in Elastic Regions.  

Eyal Oren, Spyros Kotoulas, George Anadiotis, Ronny Siebes, Annette ten Teije, and Frank van Harmelen.  
Marvin: Distributed reasoning over large-scale Semantic Web data.  

Xian-He Sun and Lionel M. Ni.  
Scalable Problems and Memory-Bounded Speedup.  

Ramakrishna Soma and V. K. Prasanna.  
Parallel Inferencing for OWL Knowledge Bases.  

Jacopo Urbani, Spyros Kotoulas, Jason Maassen, Frank van Harmelen, and Henri Bal.  
OWL reasoning with WebPIE: calculating the closure of 100 billion triples.  
Conclusion

