The Three-Weight Algorithm
A Flexible Platform for Integrating Knowledge and Optimization

Nate Derbinsky

with: José Bento, Jonathan S. Yedidia

Disney Research Boston

Architecture Implementation

	Sigma (Σ; Rosenbloom 2011)	Soar (Laird 2012)
Approach	Uniform inference over tightly coupled *factor graphs*	Hybrid ecosystem of optimized algorithms (e.g. Rete, Inv. Index)
Benefits	Speed of implementing/ evaluating architectural variants	Real-time efficiency, scalability for long and complex tasks
Challenges	Real-time decision cycle, scaling rich representations	Prototyping and evaluating architectural modifications

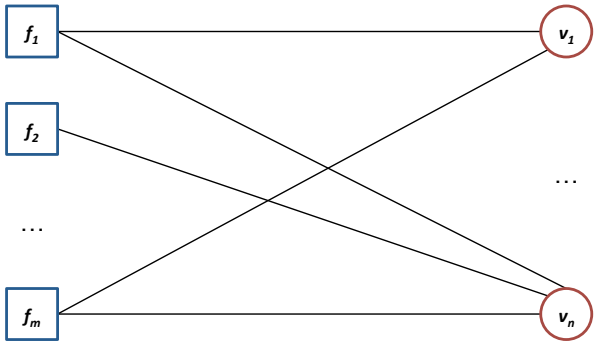
This Talk. Three-Weight Algorithm (TWA) for hybrid architecture development

- Fully general: *optimization* over factor graphs
- Efficient & scalable: distributed message-passing
- Knowledge integration: novel methods -> better expressiveness, efficiency, scaling
- Two example domains: Sudoku and circle packing
  - NOT an architecture: platform for modules and/or solving sub-problems

Optimization?

$$\underset{\boldsymbol{v}\in\mathbb{R}^n}{\text{minimize}} : f(\boldsymbol{v}) = f_1(v_1, v_2, \dots) + f_2(\dots) + \dots$$

$$+ \sum c_k(v_1, v_2, \dots) = \begin{cases} 0 & \text{constraint met} \\ \infty & \text{else} \end{cases}$$



<u>Generality</u>

Diverse processing, such as constraint satisfaction and vision/ perception

<u>Independence of Problem Specification</u>

Changing objective does *not* require changing solving method (though solution time/quality may improve with specialization)

Three-Weight Algorithm (TWA)

Message-passing algorithm (Derbinsky et al. 2013), based on Alternating Direction Method of Multipliers (ADMM)

General

- Arbitrary objective functions, constraints, and variables
- Global minimum for convex problems
- If converges, produces a feasible solution (all hard constraints met)

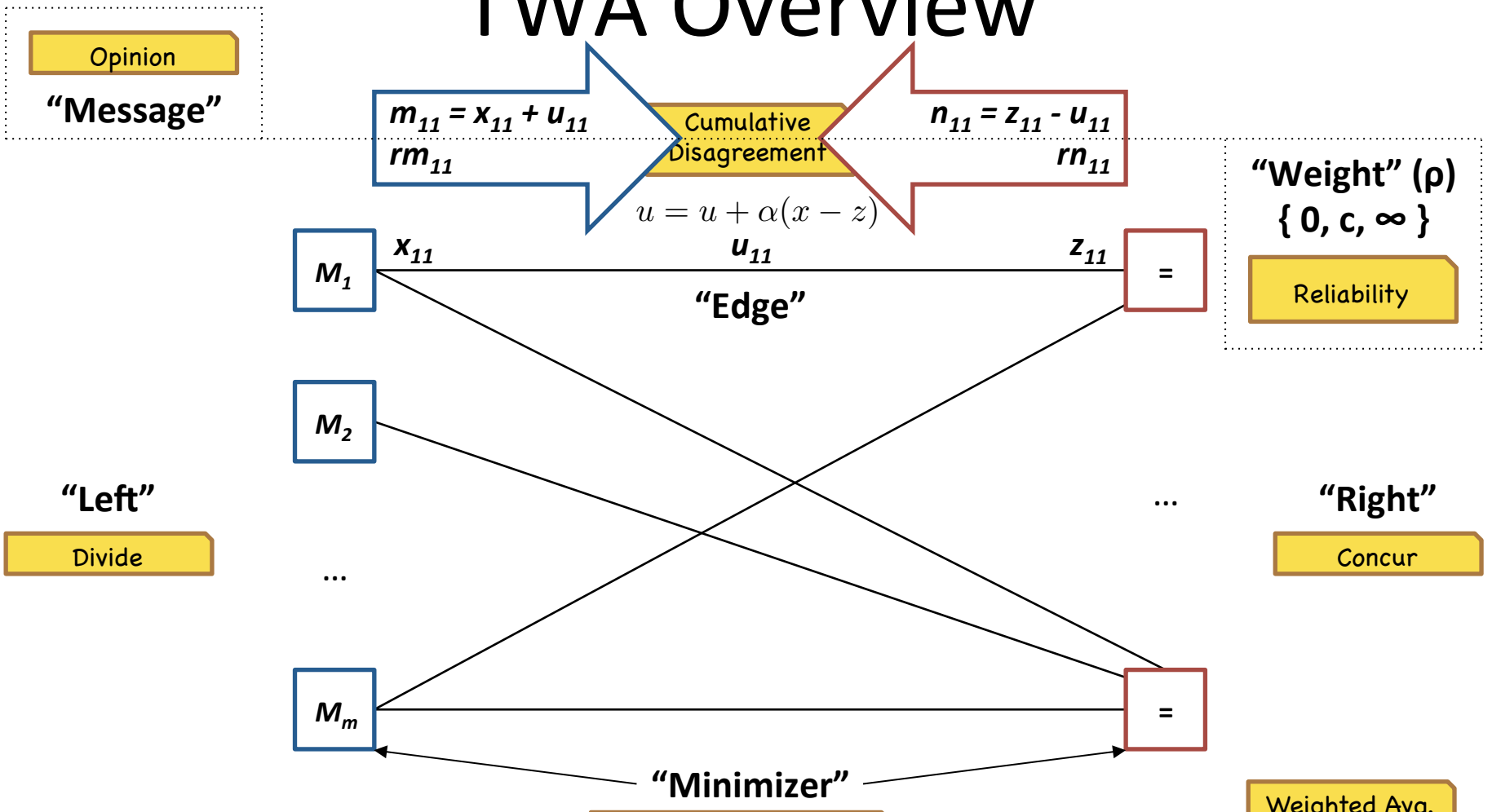
Interruptible

- Iterative algorithm; intermediate results can serve as heuristic start for complementary approaches

Scalable and Parallelizable

- Formulated as a *decomposition-coordination* problem; leads naturally to concurrency at multiple levels (e.g. MapReduce, multi-core, GPU)

TWA Overview

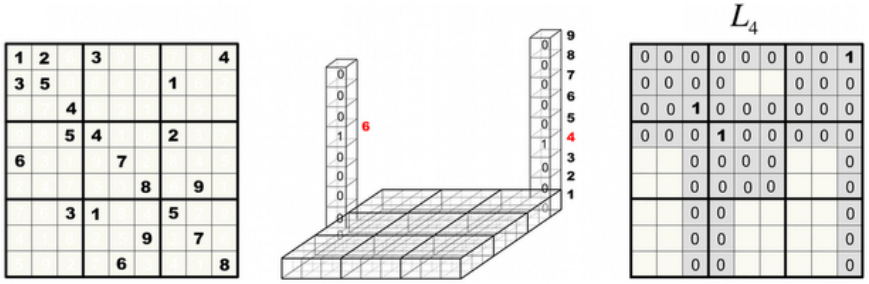


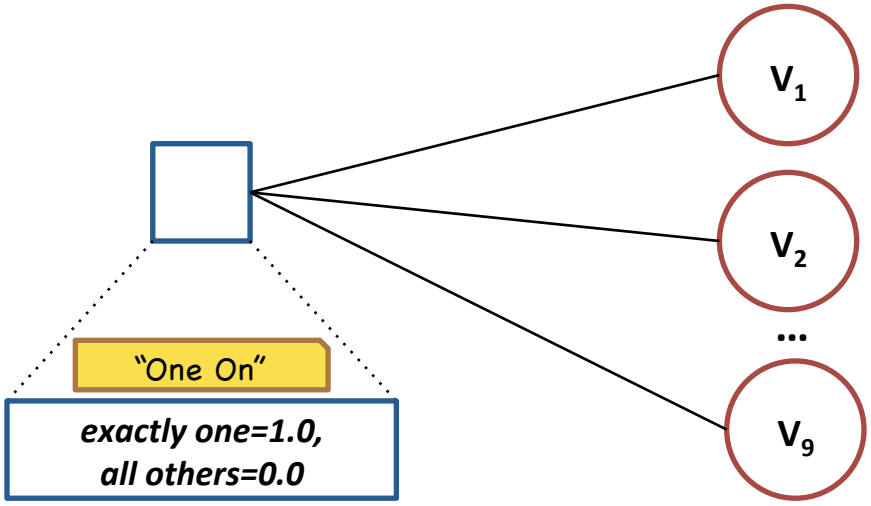
Knowledge Integration

	Reasoner Hierarchy	Graph Dynamics
Integration	Local = special factor node. Global = post-iteration logic. Scalable network to extract state & inject knowledge.	Supports changing graph topology & parameterization during execution ala changes in environment, task, or agent knowledge/preferences.
Expressiveness	Supports relational reasoning & rich representations (e.g. rules, perceptual primitives).	Supports dynamic variable sets w/o exhaustive enumeration.
Efficiency & Scaling	Local = parallelized -> discrimination network.	Smaller graph size = faster inner-loop iteration time.

Task 1: Sudoku

All hard constraints, discrete variables





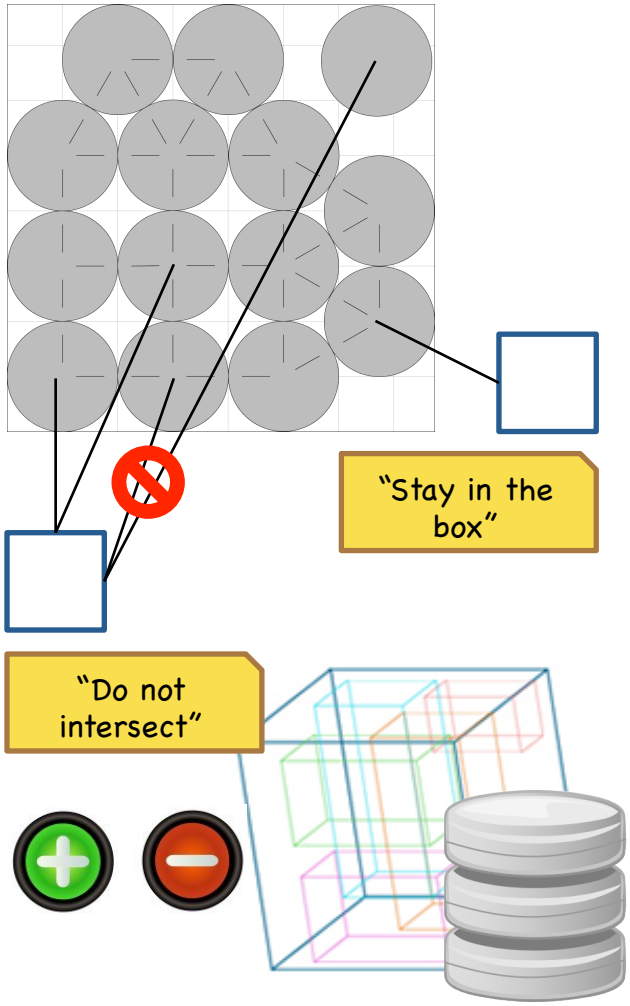
Knowledge Integration

- Local reasoners: 1-per-cell, discrimination network of possibility-set *changes*
- Global reasoner: remove 4 problem-graph edges per change; remove up to 4 factors and 1 variable
- Results: >10x reduction in graph size and avg. solve time; maintains <11msec avg. iteration for 49x49 on a single core (>125k graph nodes); close to linear speedup with added cores for large problems

SUDOKU DEMO!

Task 2: Circle Packing

All hard constraints, continuous variables



Knowledge Integration

- Global reasoner #1: integrate r-tree to maintain small problem graph
- Global reasoner #2: on-demand: identify circles with greatest overlap (bi-product of #1)
- Local reasoner: relays positioning messages to circles selected by (#2) or a human assistant
- Results: record-breaking packing for many circles (>2M); human assistance to consistently match records on smaller instances

CIRCLE PACKING DEMO!

Evaluation

Nuggets

- General, efficient, scalable algorithm for optimization
- Methods for integrating high-level knowledge within message-passing
- Yields state-of-the-art performance on numerous problems

Coal

- Have yet to…
  - Integrate with an agent/ architecture
  - Demonstrate learning