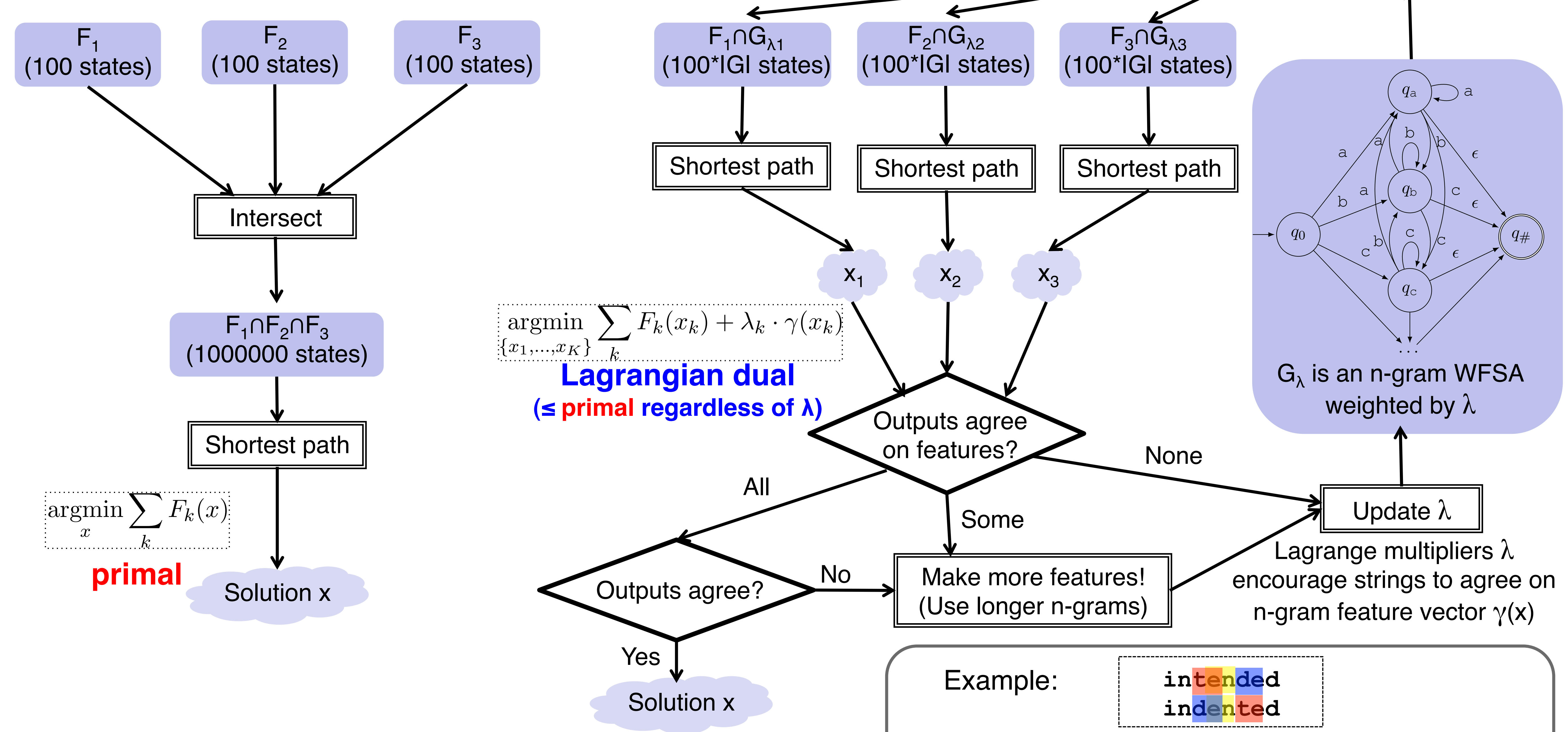
Implicitly Intersecting Weighted Automata using Dual Decomposition

Michael J. Paul and Jason Eisner
Johns Hopkins University

Solving for Sequences with Dual Decomposition

- There are many problems in NLP where we want to identify the **best string or sequence** under a model with constraints.
  - Sequence labeling, MT, transliteration, ASR, consensus finding, etc.
- Often we use dynamic programming for this, but this can be **exponentially expensive** in the number of constraints.
- **Weighted finite state automata (WFSAs)** are a fairly general encoding of how well strings are scored under a model. The **intersection** of two or more WFSAs yields a WFSA which accepts strings with the **combined score** across all of the WFSAs.
- Intersection is expensive – intersections produce automata whose size increases **multiplicatively** rather than additively.
- **Dual decomposition** allows us to reformulate this so that we find the best string in each WFSA independently, rather than the best string in their full intersection – now we are back to an **additive** cost.
- This decomposition is valid if we add **agreement constraints** requiring that each of the WFSAs output the same string.
- Solving for each WFSA under these constraints can still be done efficiently and independently if we **relax the constraints** with Lagrange multipliers.
- We solve the **Lagrangian dual** with a subgradient ascent algorithm. The dual function is convex and lower bounds the primal, thus if we converge to a feasible solution in the dual, we know we have found the **global optimum** of the primal.
- Future work: MAP inference in **graphical models over strings** (Dreyer&Eisner 2009).

Explicit Intersection vs Dual Decomposition

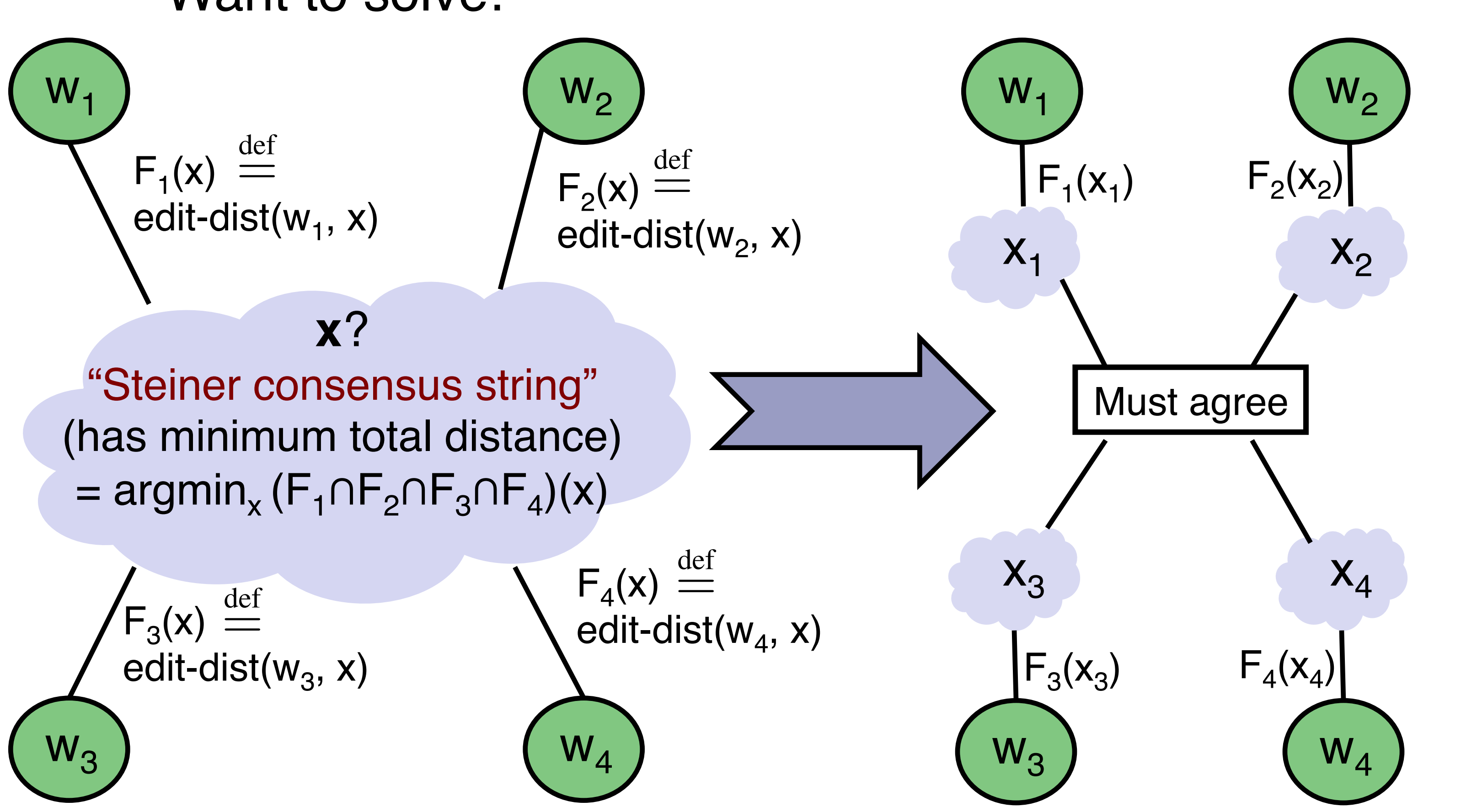


Example: intended / indented

- Agree on counts of 2-grams '`en`', '`te`', '`de`'.
- But disagree on counts of 3-grams '`ten`', '`end`', '`den`', etc.
  - So add these 3-grams as features!

Experiments with Consensus Decoding

Want to solve:



This is a hard problem to solve exactly.
We show that we can often do it anyway with dual decomposition.

Iterations	Outputs $x_1, \ldots, x_5$ at current iteration
0	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> WE WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I DON'T WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> WELL I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> THEY WANT TO BE TAKING A DEEP BREATH NOW </s>
300	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> WE WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I DON'T WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> WELL I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> WELL WANT TO BE TAKING A DEEP BREATH NOW </s>
375	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I DON'T WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
472	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>
	<s> I WANT TO BE TAKING A DEEP BREATH NOW </s>

Above: An example of 5 (of 25) strings output at various iterations of our algorithm, where convergence is reached after 472 iterations. We performed consensus decoding on the top 25 strings from 226 lattices produced by the IBM Attila decoder on Broadcast News utterances. 85% of the problems converged to an exact solution within 1000 iterations.

We also synthesized many different types of consensus problems to see how our algorithm behaved. For each problem, we generated a random base string, then generated K random mutations of the string. We adjusted the number of strings *K*, the length of the base string ℓ, the alphabet size $|\Sigma|$, and the mutation probability μ.

Right: Across 100 random trials, we show the percentage which converged within 1000 iterations, the average number of iterations, and the reduction in score from the starting point.
Bottom: Typical plots of the primal score (original objective) and the smaller dual score (the current Lagrangian objective). The algorithm converges when the duality gap is closed.

K	ℓ	$\lvert\Sigma\rvert$	μ	**Conv.**	**Iters.**	**Red.**
5	100	5	0.1	68%	257 (±110)	24%
5	100	5	0.2	0%	–	8%
5	50	5	0.1	80%	123 (± 65)	20%
5	50	5	0.2	10%	436 (±195)	18%
10	50	5	0.1	69%	228 (±164)	18%
10	50	5	0.2	0%	–	8%
10	50	5	0.4	0%	–	3%
10	30	10	0.1	100%	50 (± 69)	13%
10	30	10	0.2	93%	146 (±142)	20%
10	30	10	0.4	0%	–	16%
10	15	20	0.1	100%	26 (± 6)	1%
10	15	20	0.2	98%	43 (± 18)	10%
10	15	20	0.4	63%	289 (±217)	18%
10	15	20	0.8	0%	–	11%
25	15	20	0.1	98%	30 (± 5)	0%
25	15	20	0.2	92%	69 (±112)	6%
25	15	20	0.4	55%	257 (±149)	16%
25	15	20	0.8	0%	–	12%
50	10	10	0.2	68%	84 (±141)	0%
50	10	10	0.4	21%	173 (± 94)	9%
100	10	10	0.2	44%	147 (±220)	0%
100	10	10	0.4	13%	201 (±138)	6%