

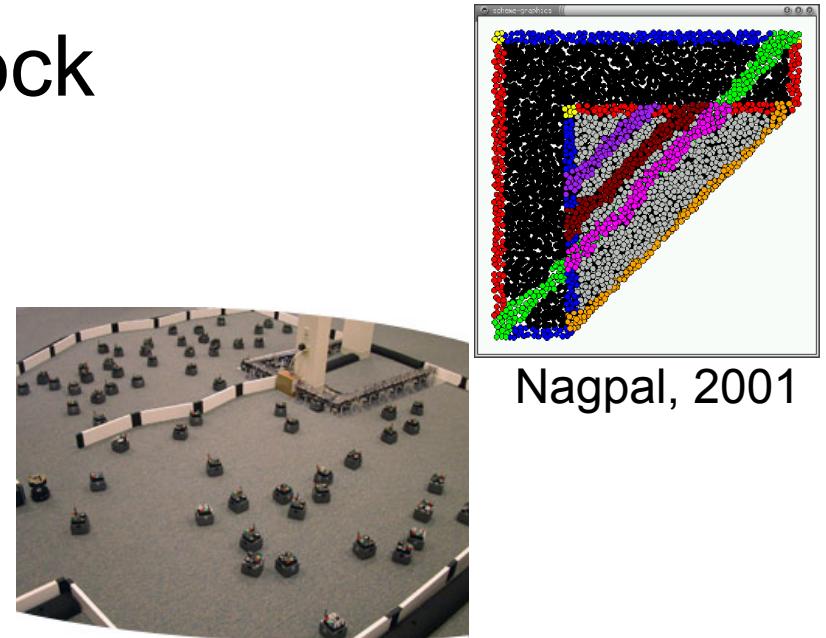
Flexible Self-Healing Gradients

Jacob Beal
BBN Technologies
ACM SAC, March 2009

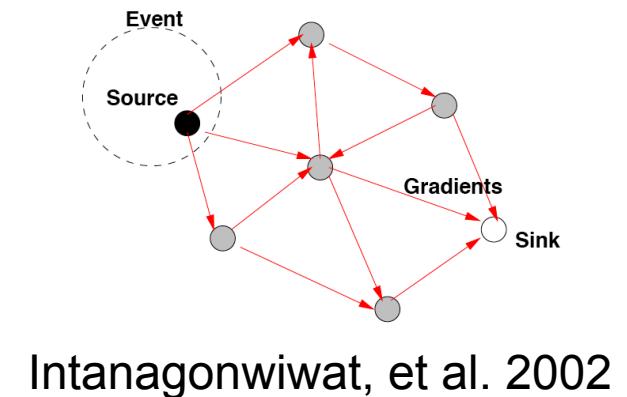
“Gradient”: Local Calculation of Shortest-Distance Estimates

Common SA/SO building block

- Pattern Formation
 - Nagpal, Coore, Butera
- Distributed Robotics
 - Stoy, Werfel, McLurkin
- Networking
 - DV routing, Directed Diffusion

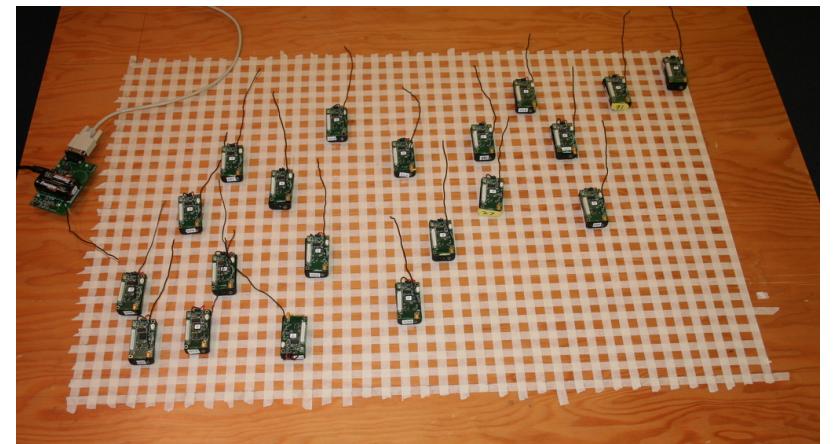


Need to adapt to changes



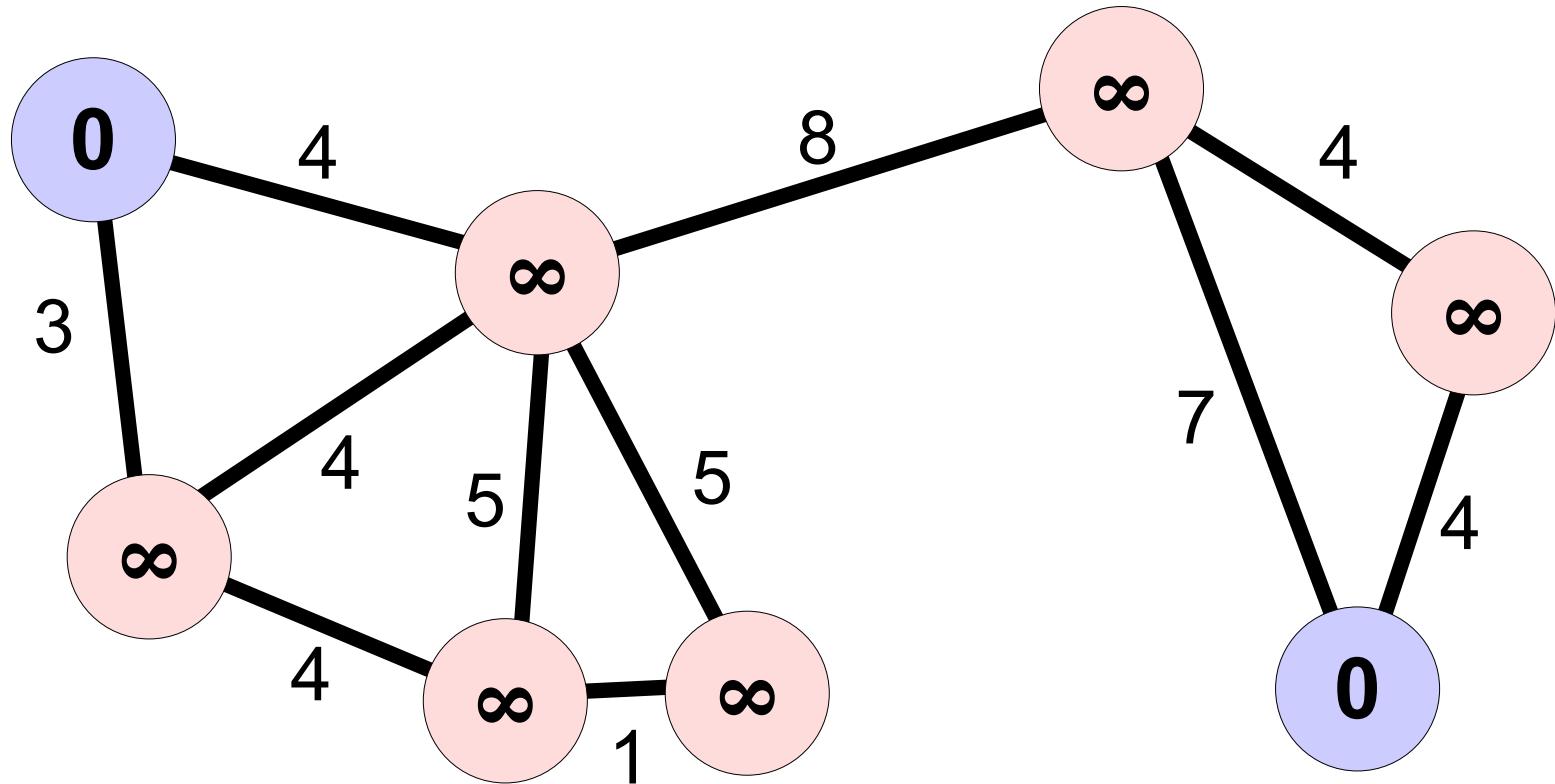
Previous Self-Healing Gradients

- “Invalidate and Rebuild”
 - GRAB: single source, rebuild on high error
 - TTDD: static subgraph, rebuild on lost msg.
- “Incremental Repair”
 - Hopcount: Clement & Nagpal, Butera
 - CRF-Gradient



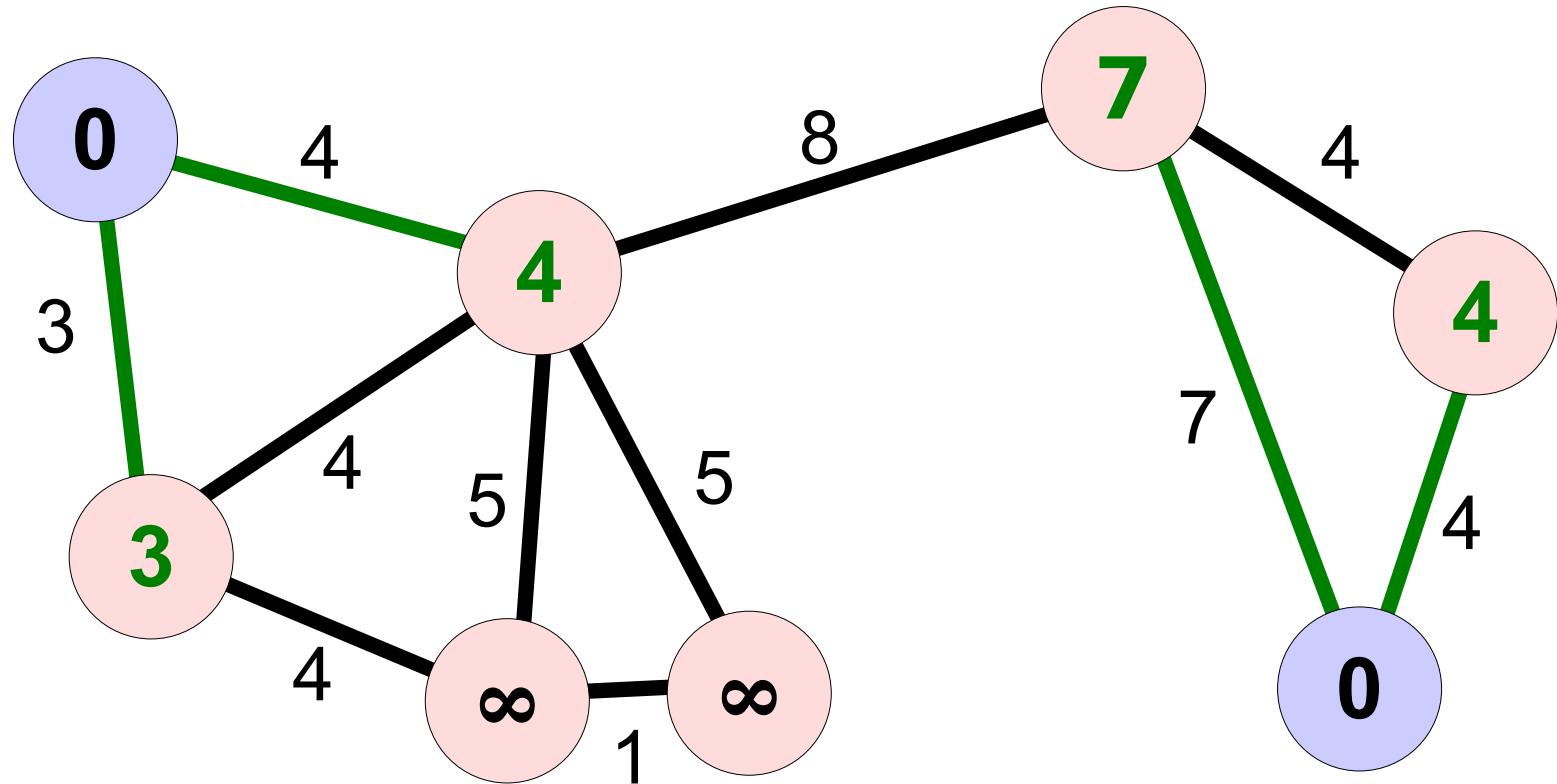
Prior work w. Mica2 Motes

Calculation By Relaxation



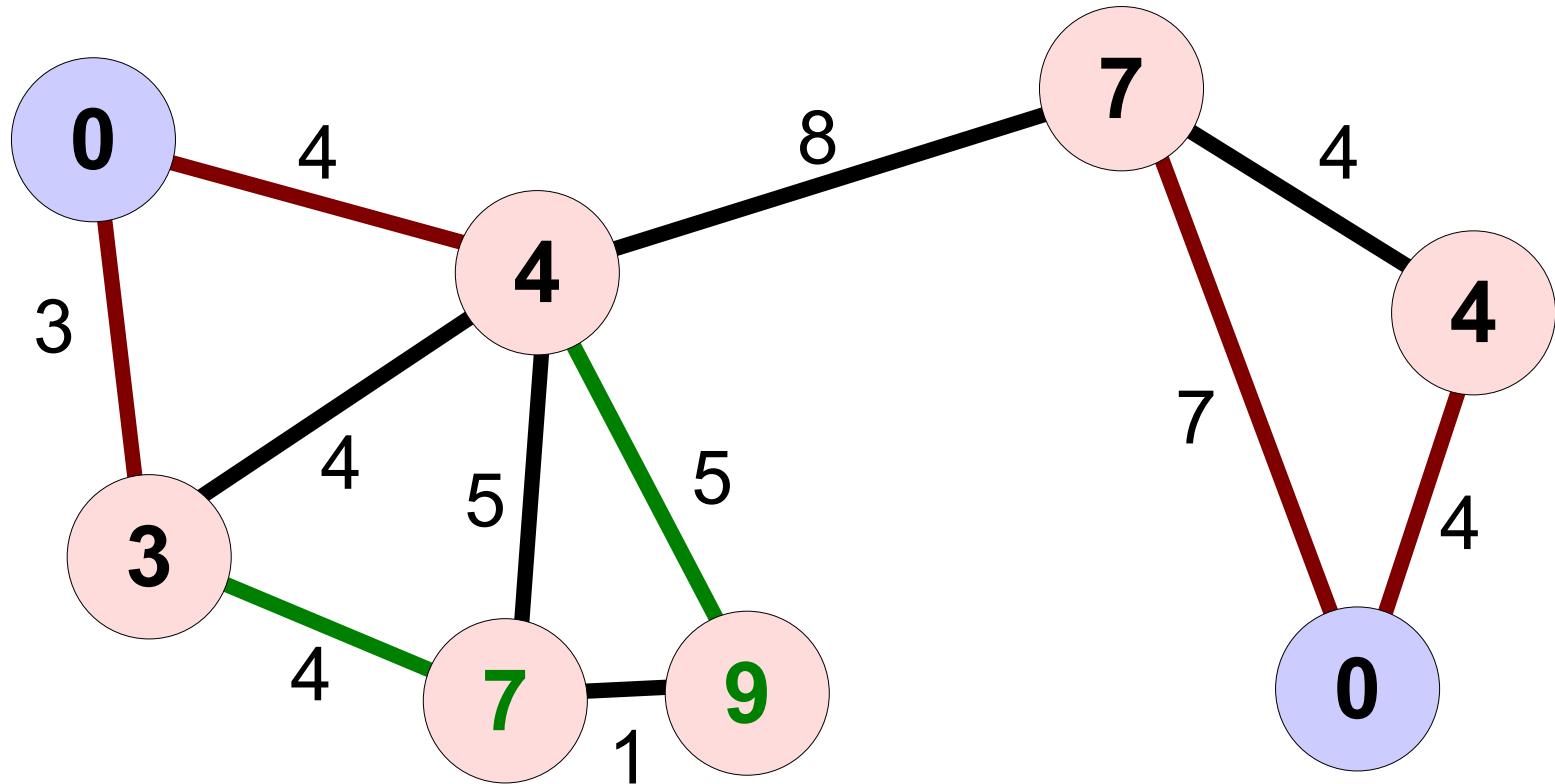
$$g_x = \begin{cases} 0 & \text{if } x \in S \\ \min \{g_y + d(x, y) \mid y \in N_x\} & \text{if } x \notin S \end{cases}$$

Calculation By Relaxation



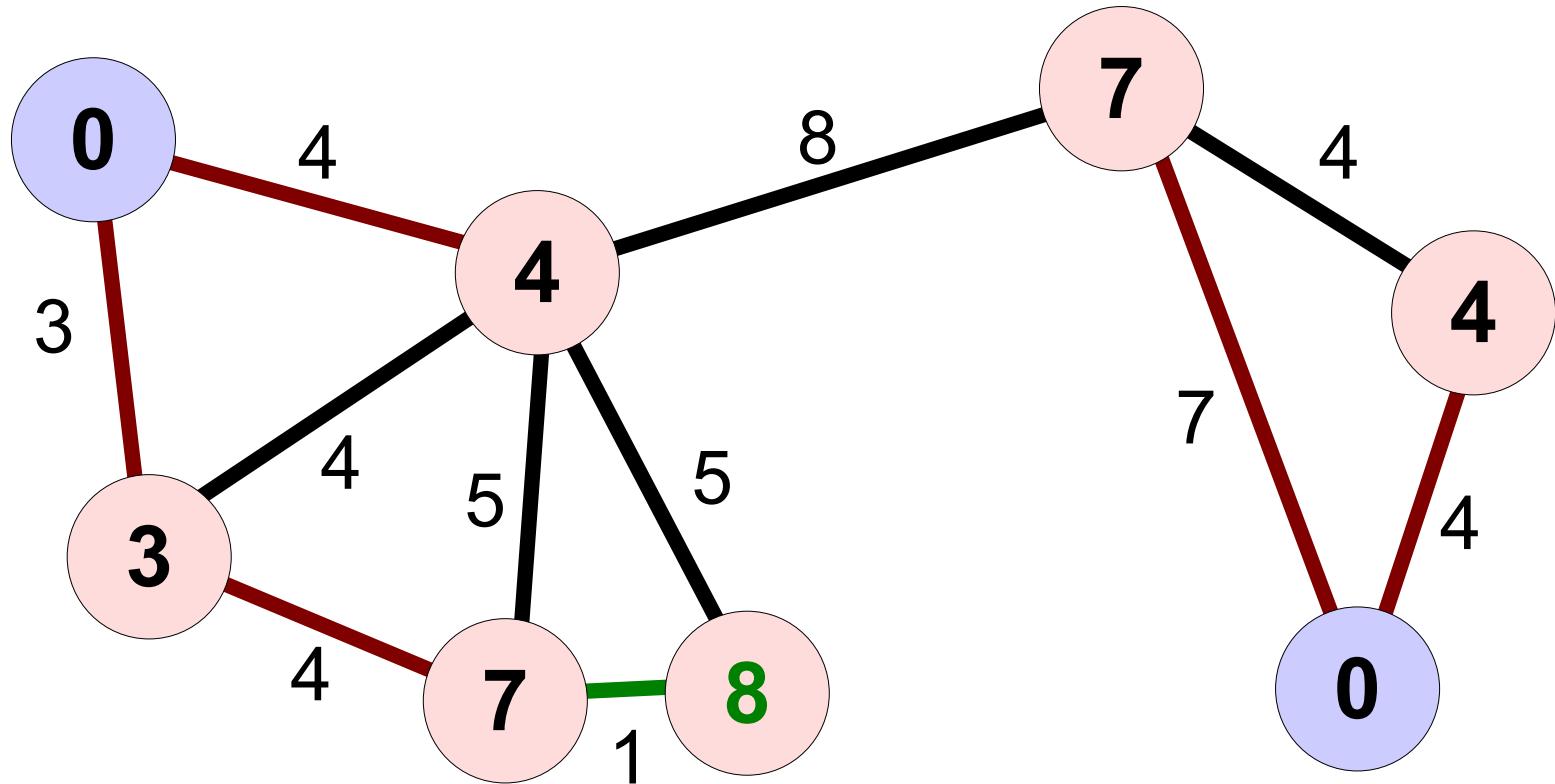
$$g_x = \begin{cases} 0 & \text{if } x \in S \\ \min\{g_y + d(x, y) \mid y \in N_x\} & \text{if } x \notin S \end{cases}$$

Calculation By Relaxation



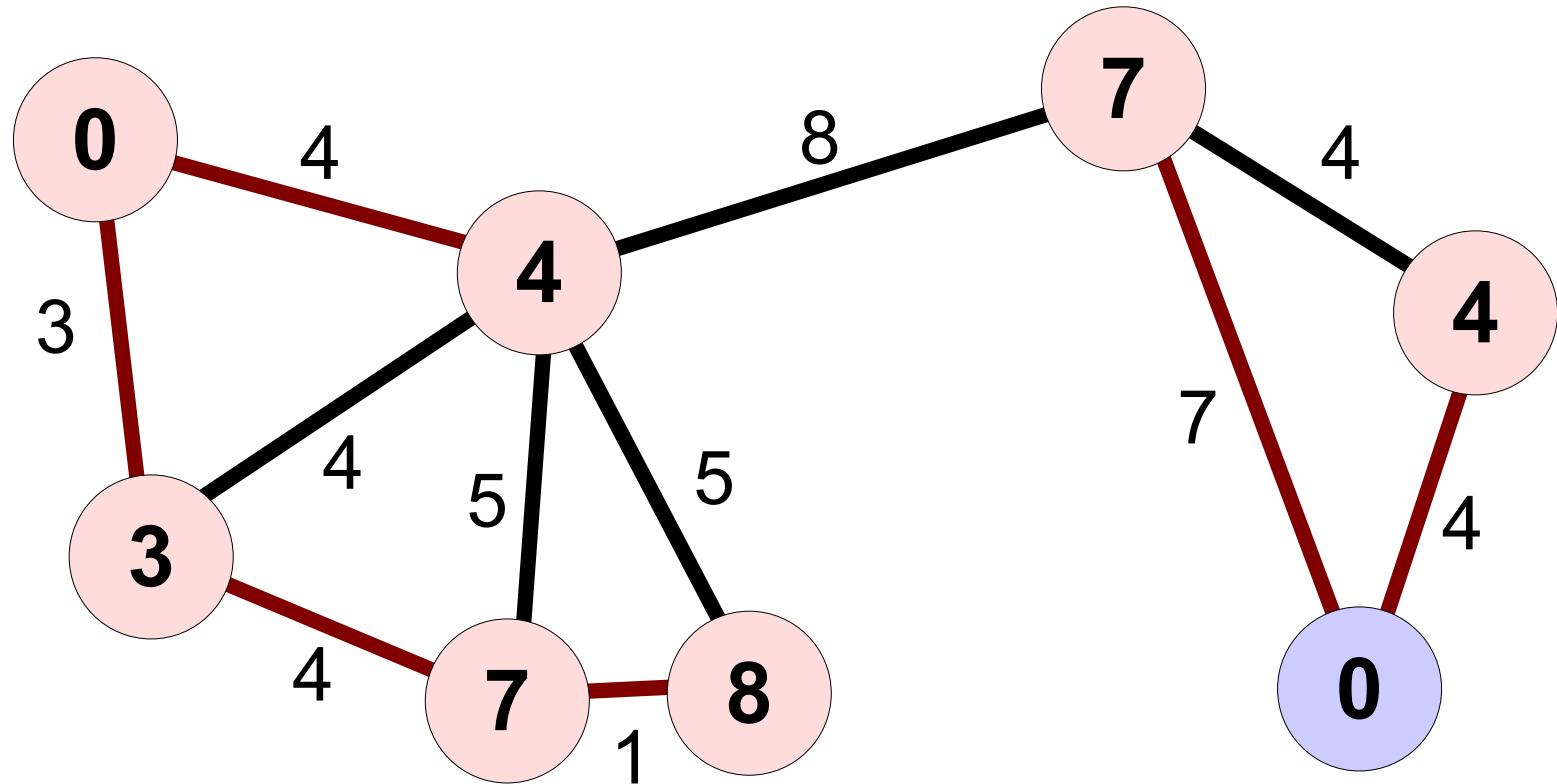
$$g_x = \begin{cases} 0 & \text{if } x \in S \\ \min\{g_y + d(x, y) \mid y \in N_x\} & \text{if } x \notin S \end{cases}$$

Calculation By Relaxation



$$g_x = \begin{cases} 0 & \text{if } x \in S \\ \min\{g_y + d(x, y) \mid y \in N_x\} & \text{if } x \notin S \end{cases}$$

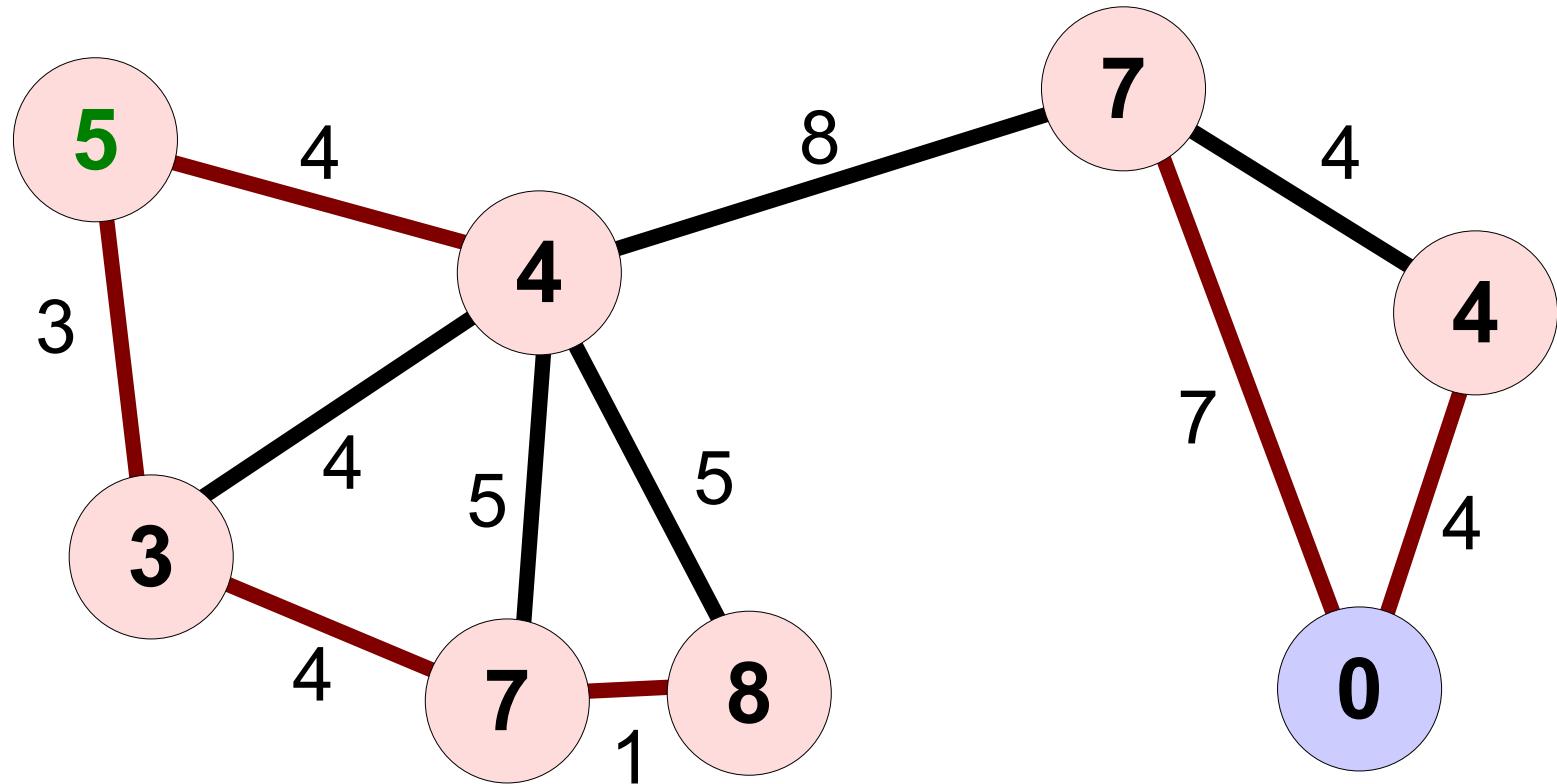
CRF Rising Values



- zero at source
- rise at v_0 with relaxed constraint
- otherwise snap to constraint

$$v_0 = 5$$

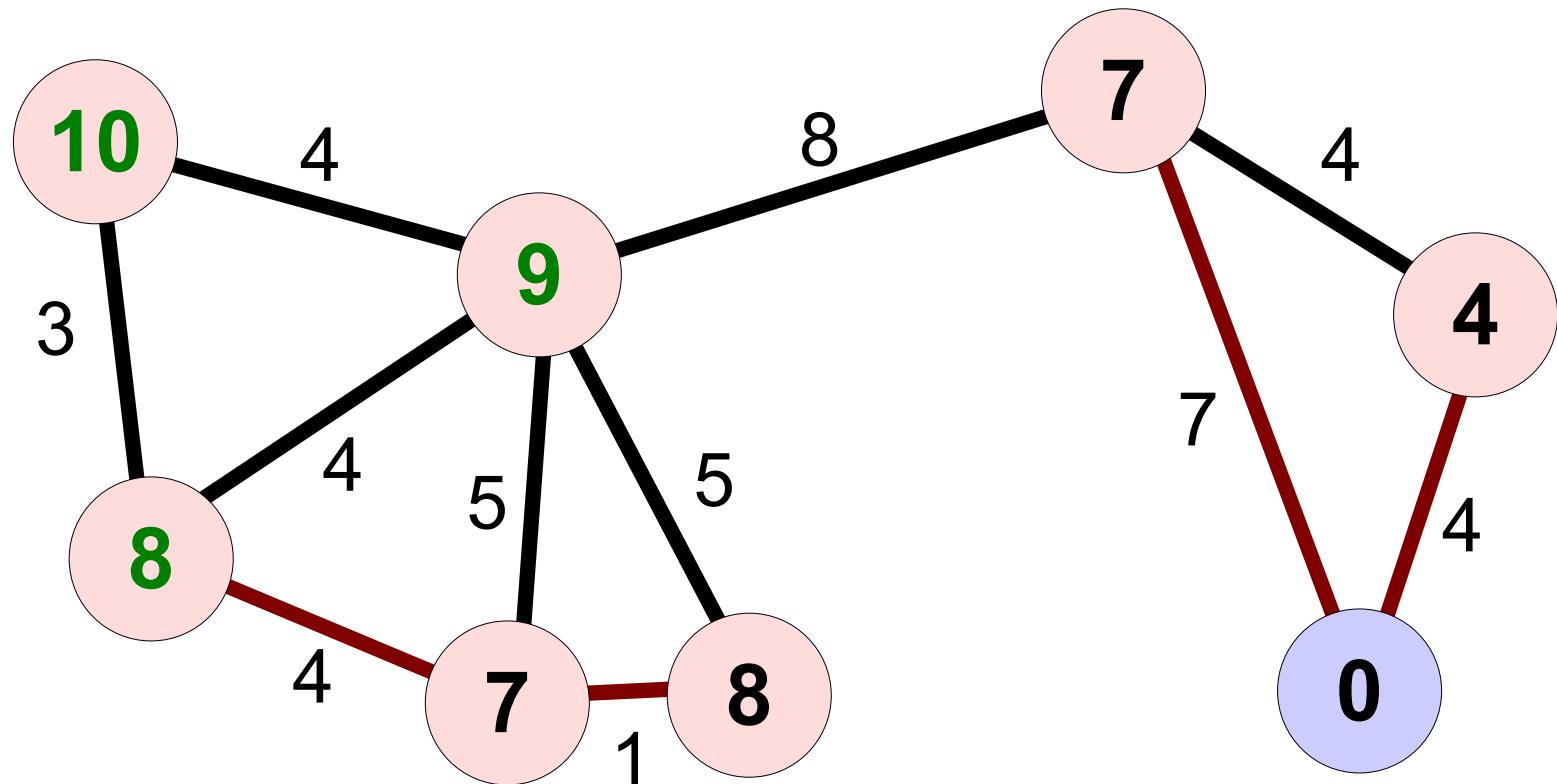
CRF Rising Values



- zero at source
- rise at v_0 with relaxed constraint
- otherwise snap to constraint

$$v_0 = 5$$

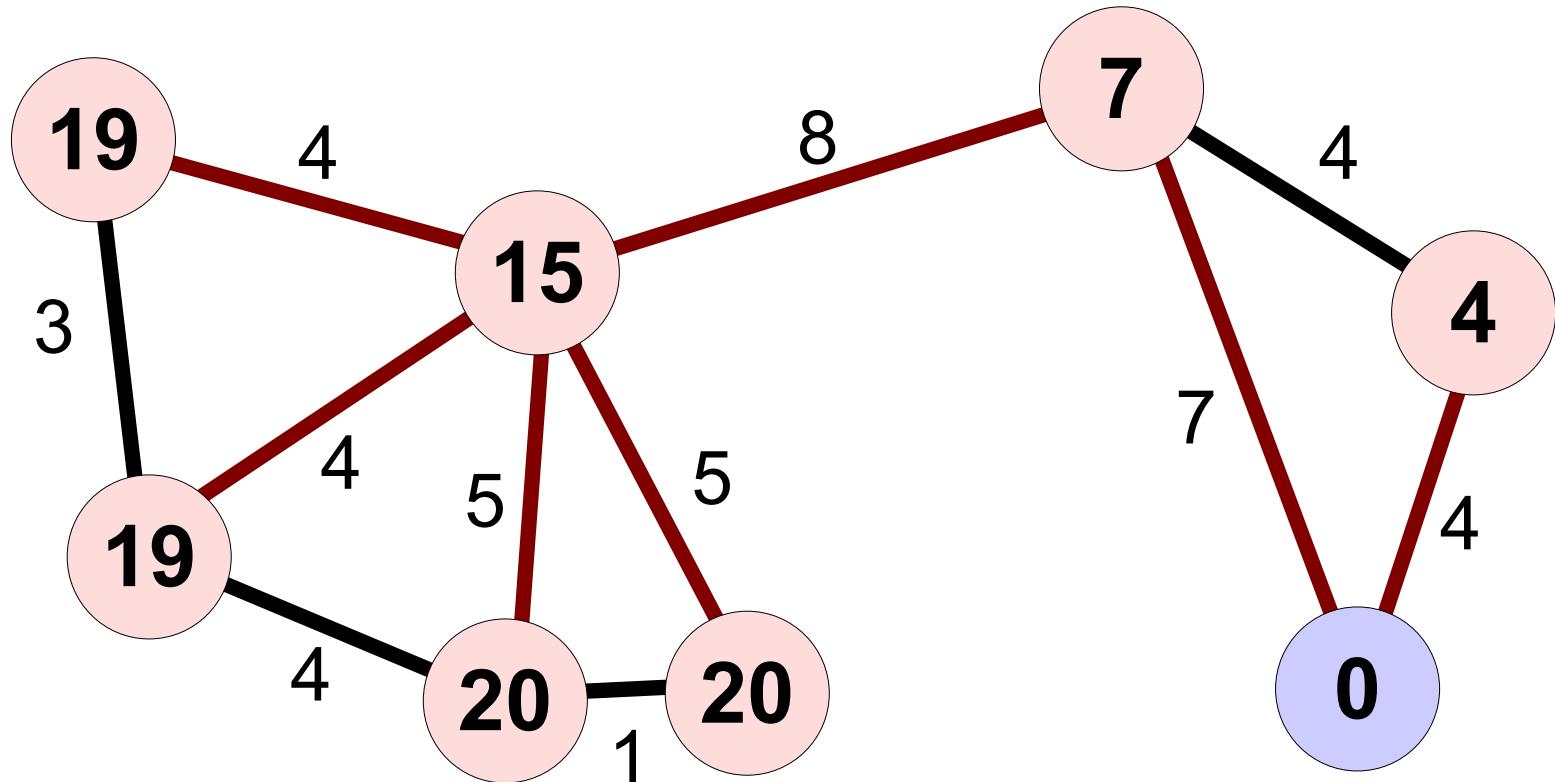
CRF Rising Values



- zero at source
- rise at v_0 with relaxed constraint
- otherwise snap to constraint

$$v_0 = 5$$

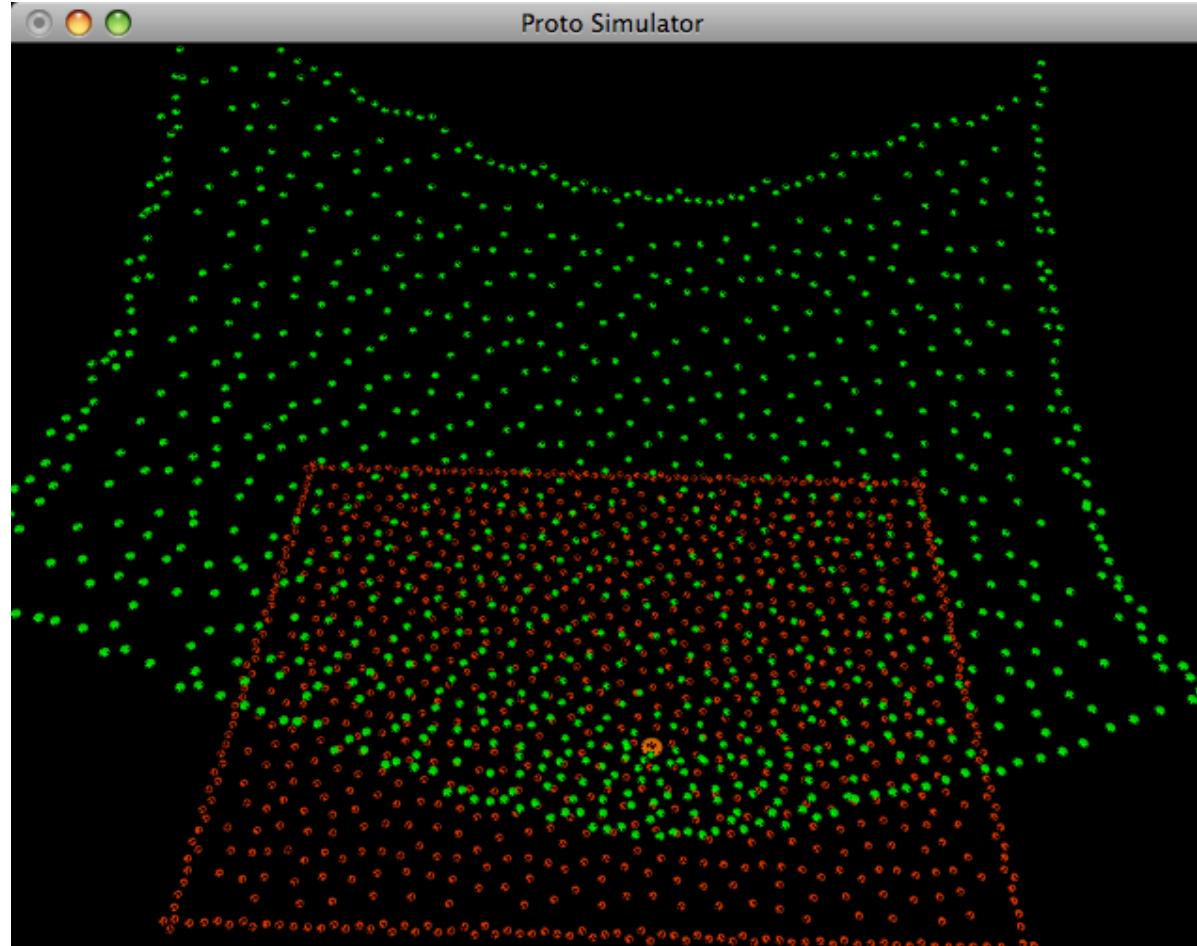
New Gradient Values



- zero at source
- rise at v_0 with relaxed constraint
- otherwise snap to constraint

$$v_0 = 5$$

Perfection is expensive and “twitchy”



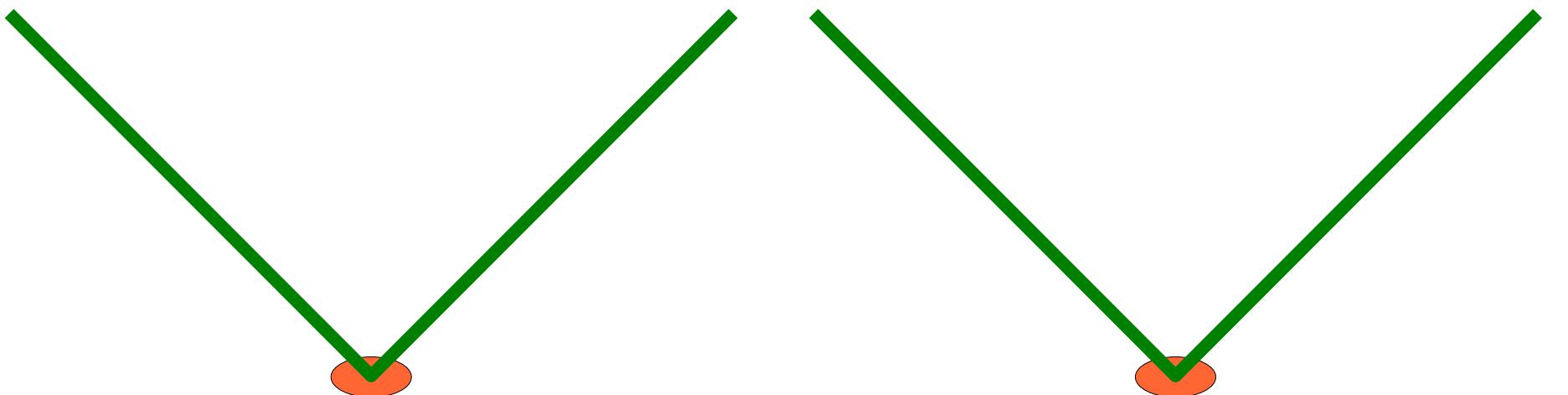
But most applications don't need perfection...

```
proto -n 1000 -r 10 "(all (mov (* 0.1 (disperse))) (green (gradient (sense 1))))" -l -s 1 -m -w
```

Making gradients tolerate error

- Hysteresis?
 - Past a threshold, unbounded communication
- Low-pass filtering?
 - Worse! Value change \neq msg cost
- “Elastic” connections!
 - Absorb error incrementally

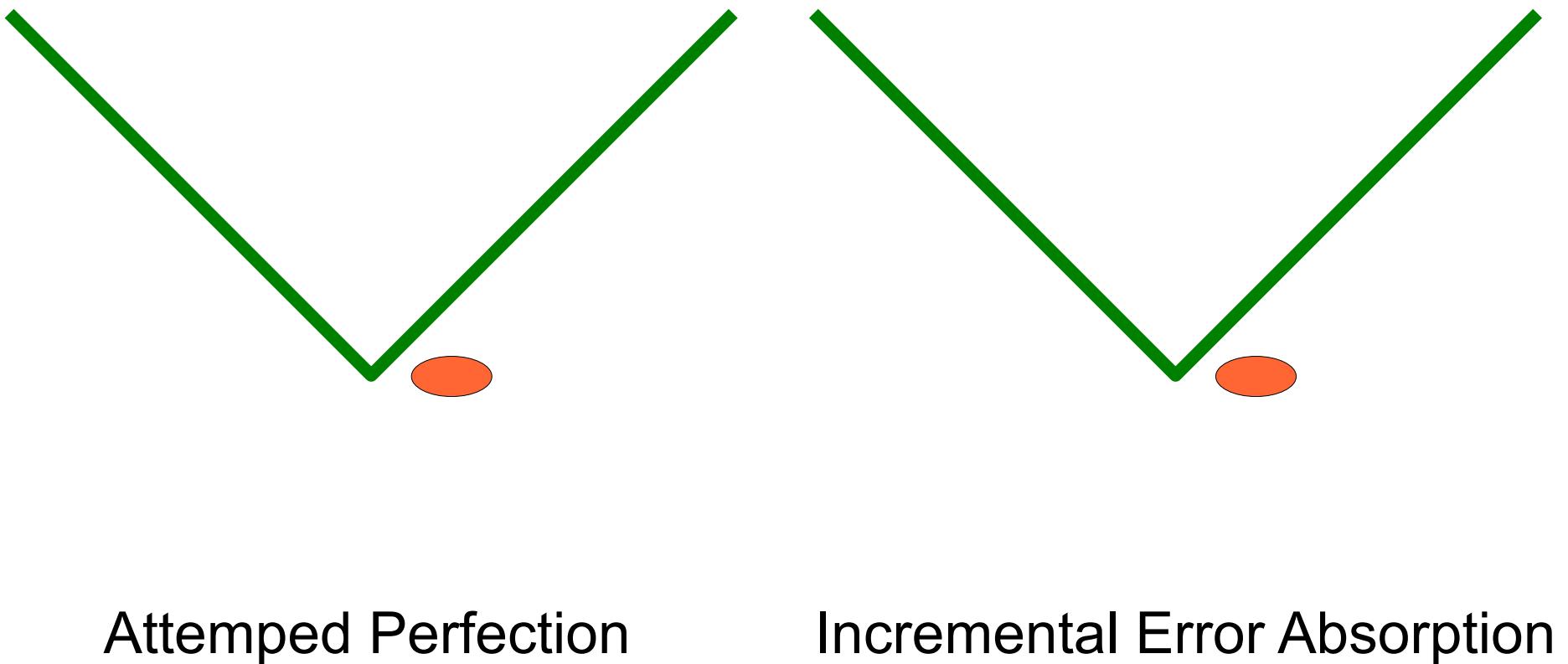
Perturbations & Absorption



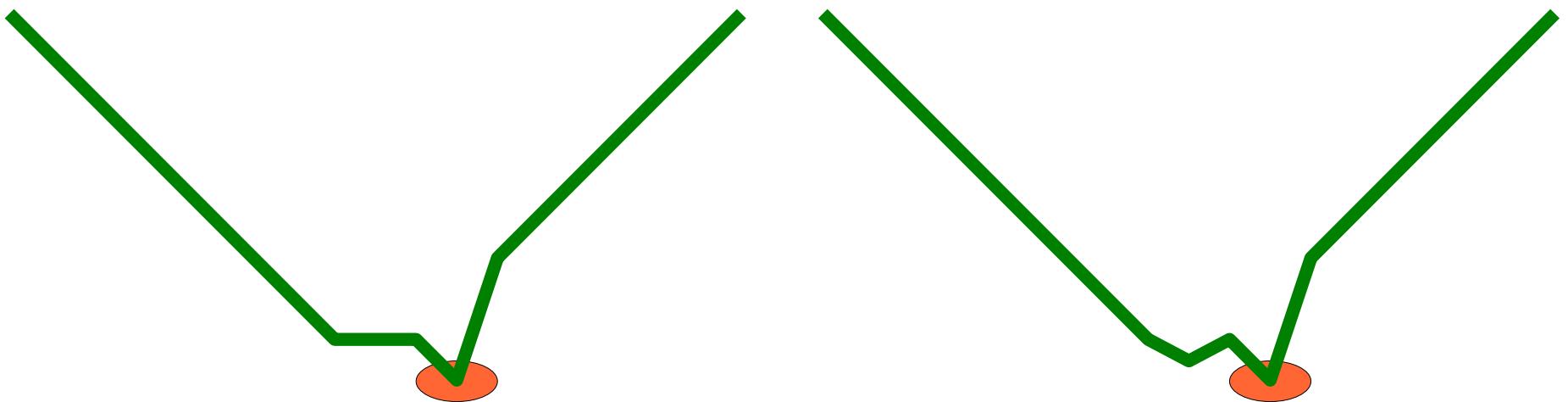
Attempted Perfection

Incremental Error Absorption

Perturbations & Absorption



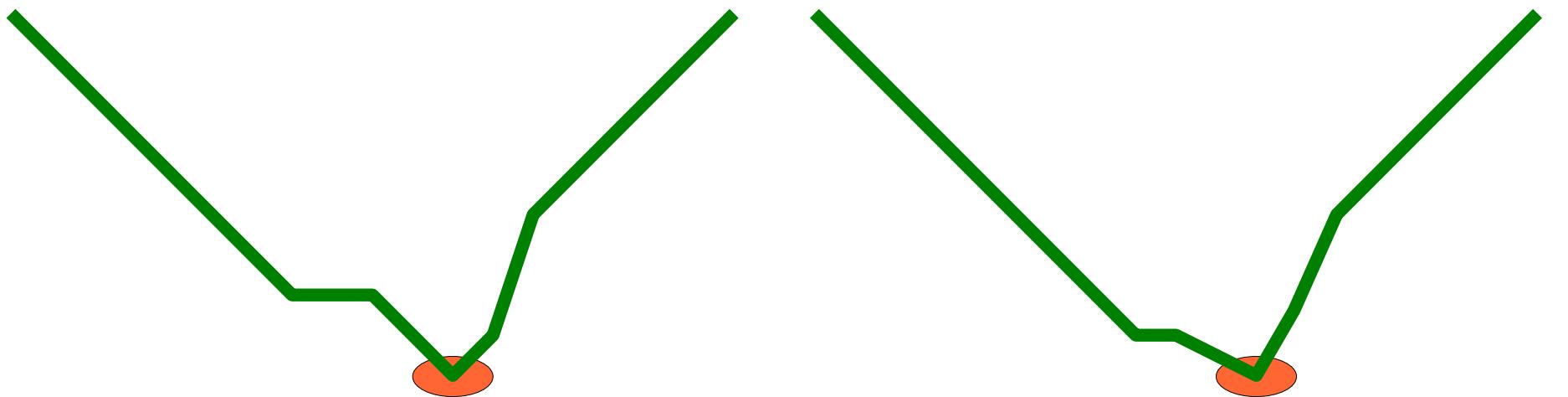
Perturbations & Absorption



Attempted Perfection

Incremental Error Absorption

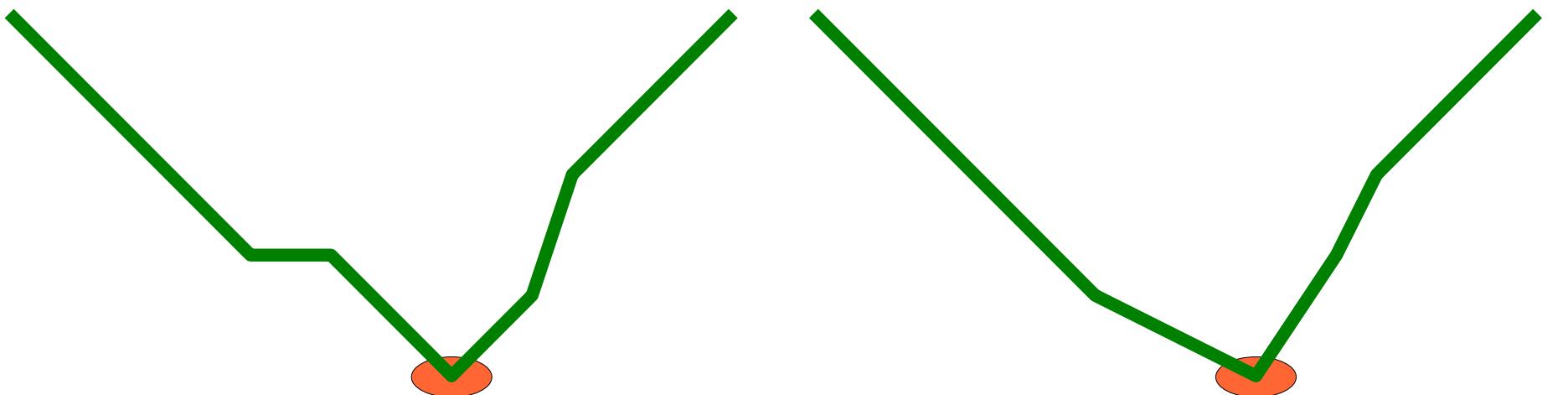
Perturbations & Absorption



Attempted Perfection

Incremental Error Absorption

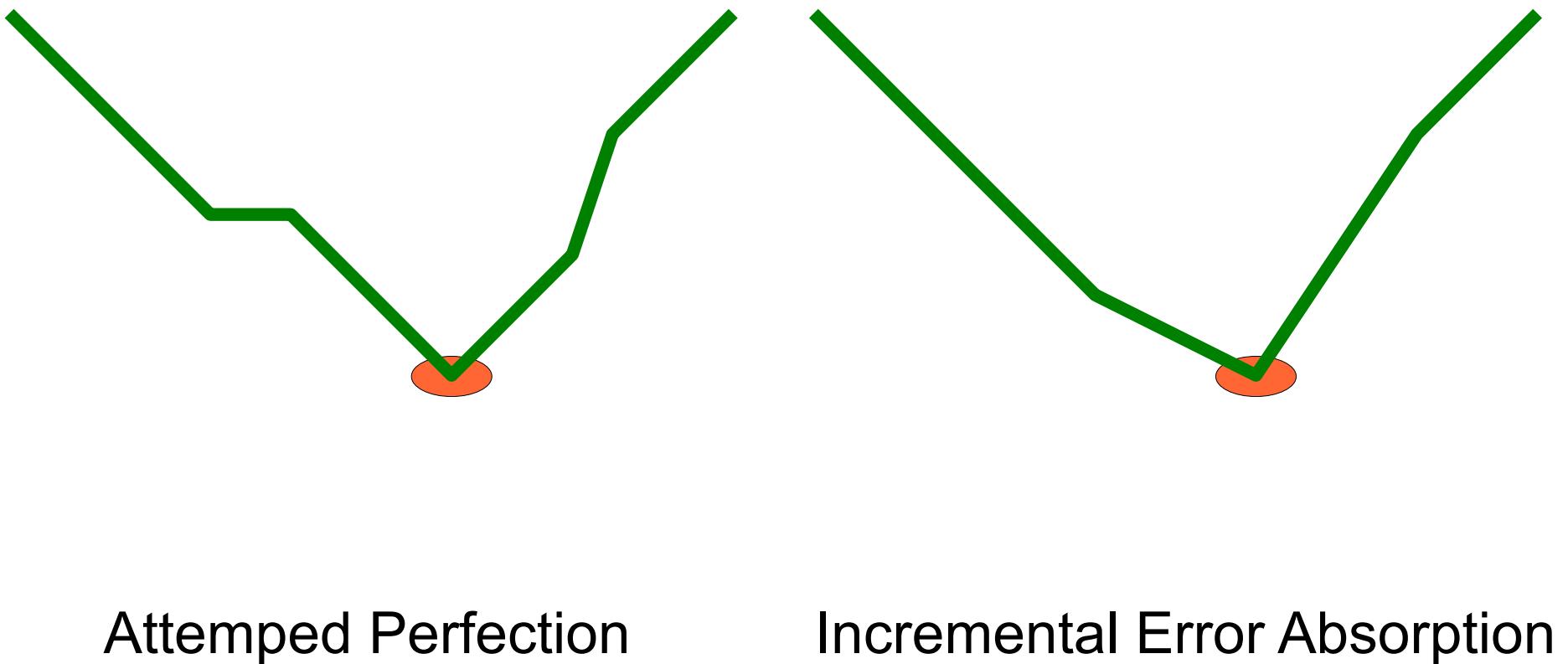
Perturbations & Absorption



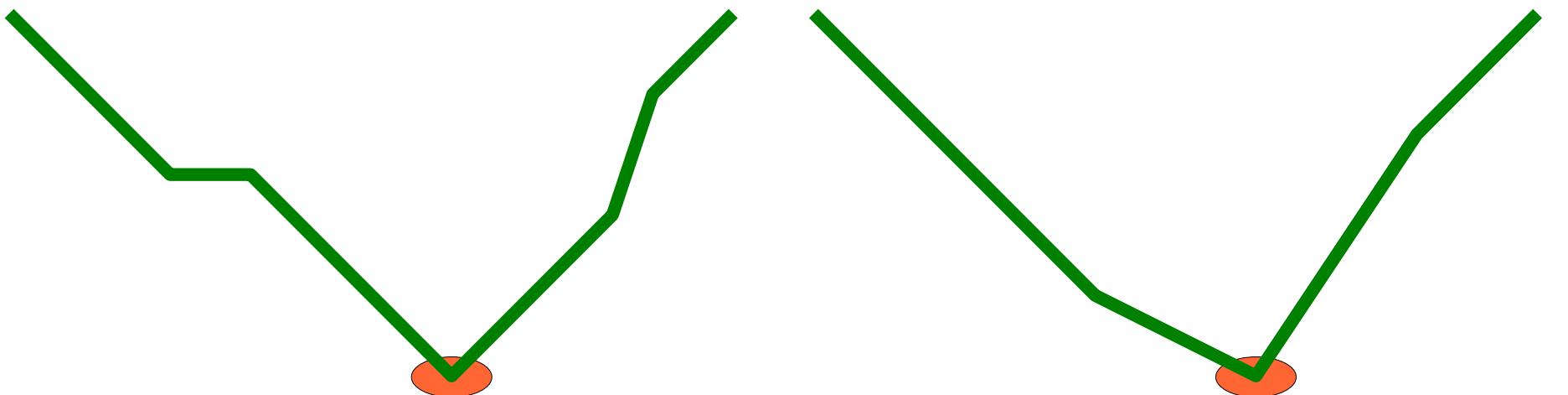
Attempted Perfection

Incremental Error Absorption

Perturbations & Absorption



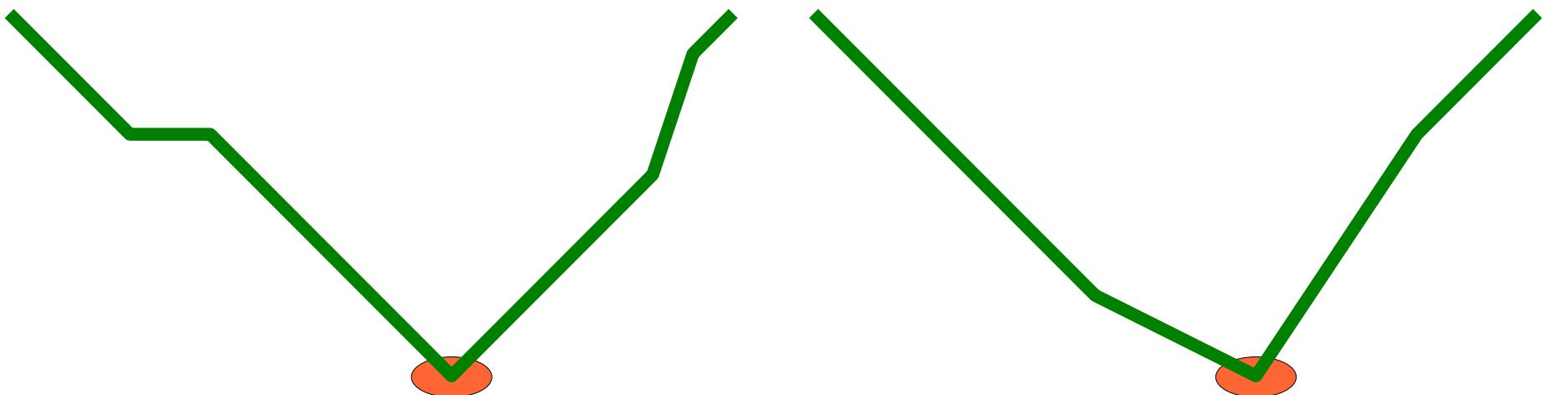
Perturbations & Absorption



Attempted Perfection

Incremental Error Absorption

Perturbations & Absorption



Attempted Perfection

Incremental Error Absorption

Managing error through slope

- Goal: ϵ -acceptable values

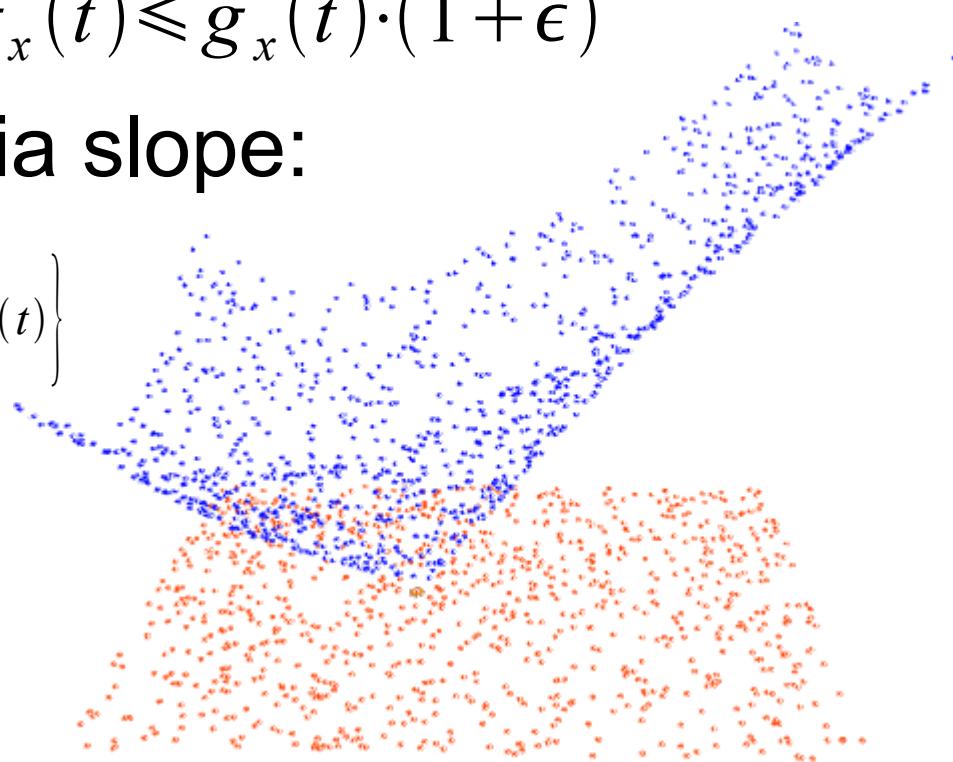
$$\bar{g}_x(t) \cdot (1 - \epsilon) \leq g_x(t) \leq \bar{g}_x(t) \cdot (1 + \epsilon)$$

- Add local constraint via slope:

$$s_x(t) = \max \left\{ \frac{g_x(t - \Delta_t) - g_y(t_{x,y})}{d(x, y, t_{x,y})} \mid y \in N_x(t) \right\}$$

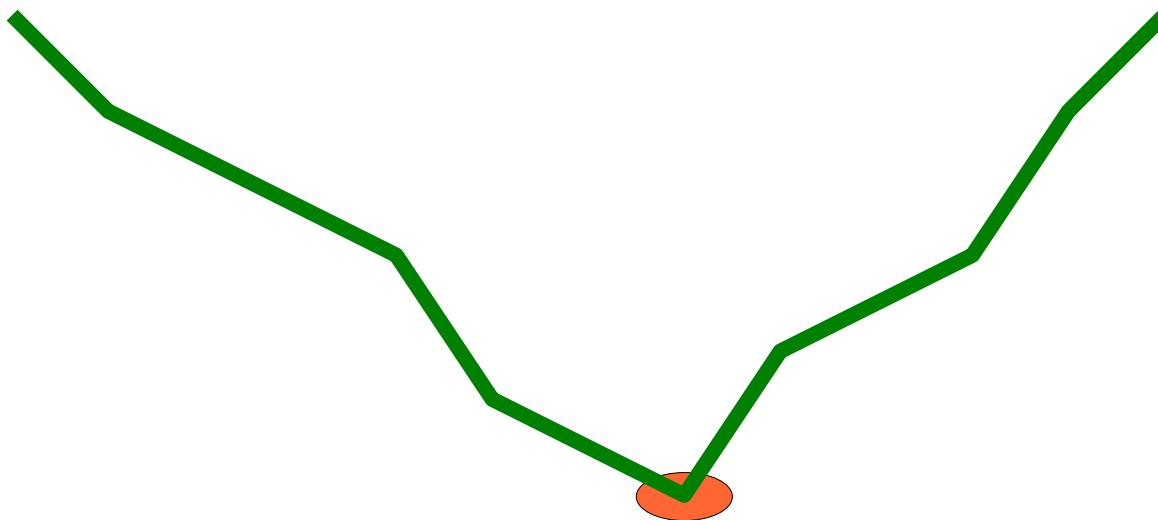
→ “flexible” gradients

(allow small distortion for rising value problem)



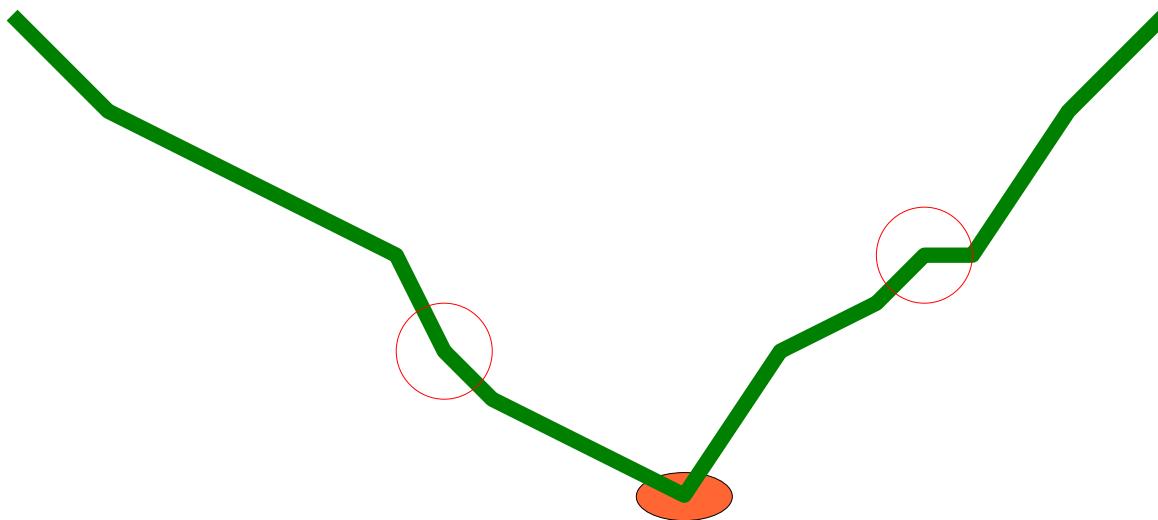
Getting the kinks out

- Flexed regions cannot absorb error
- Want eventual correctness



Getting the kinks out

- Flexed regions cannot absorb error
- Want eventual correctness

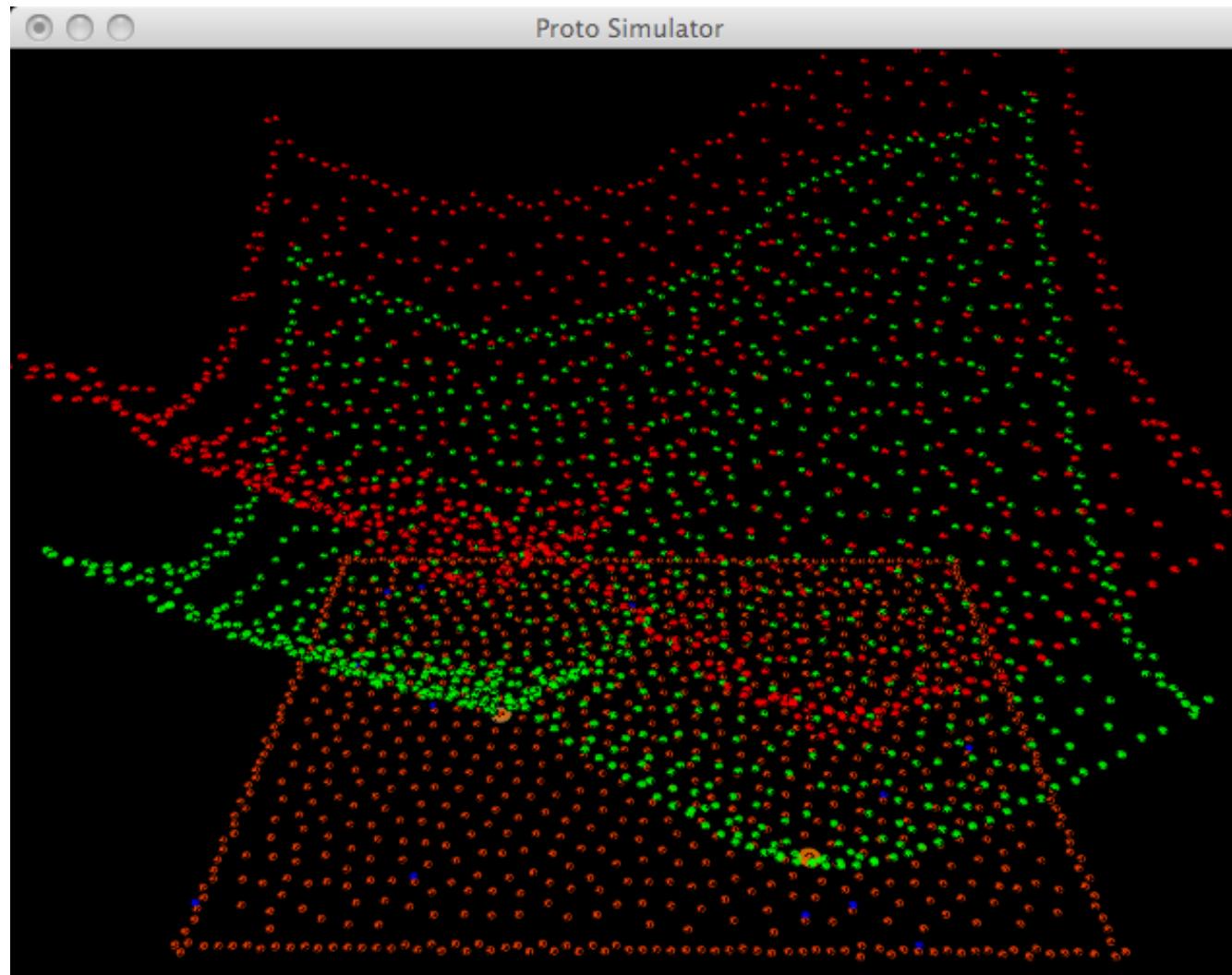


Solution: occasional $\varepsilon=0$ steps

Flex-Gradient Algorithm (simplified)

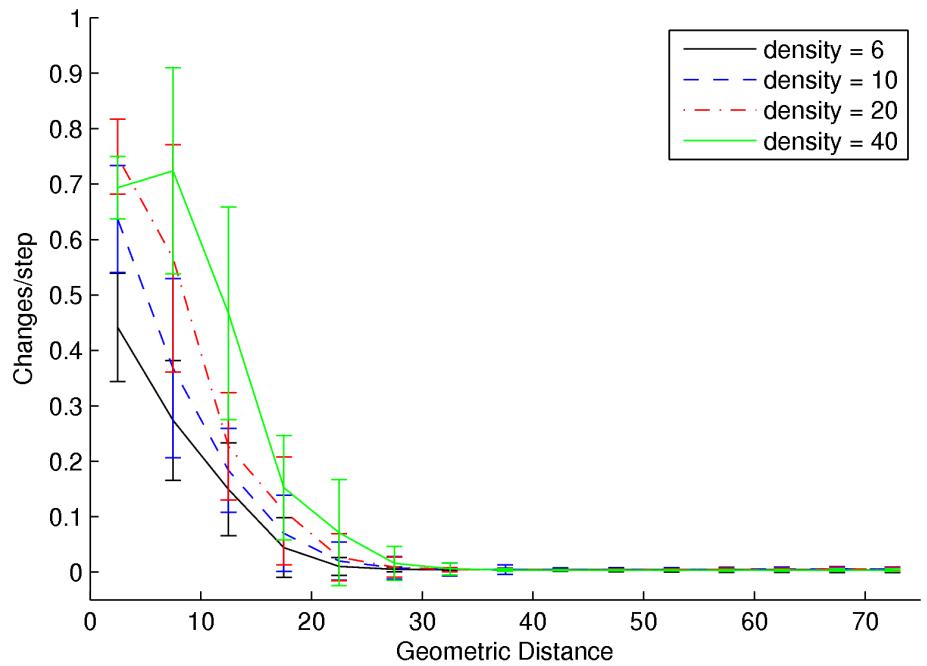
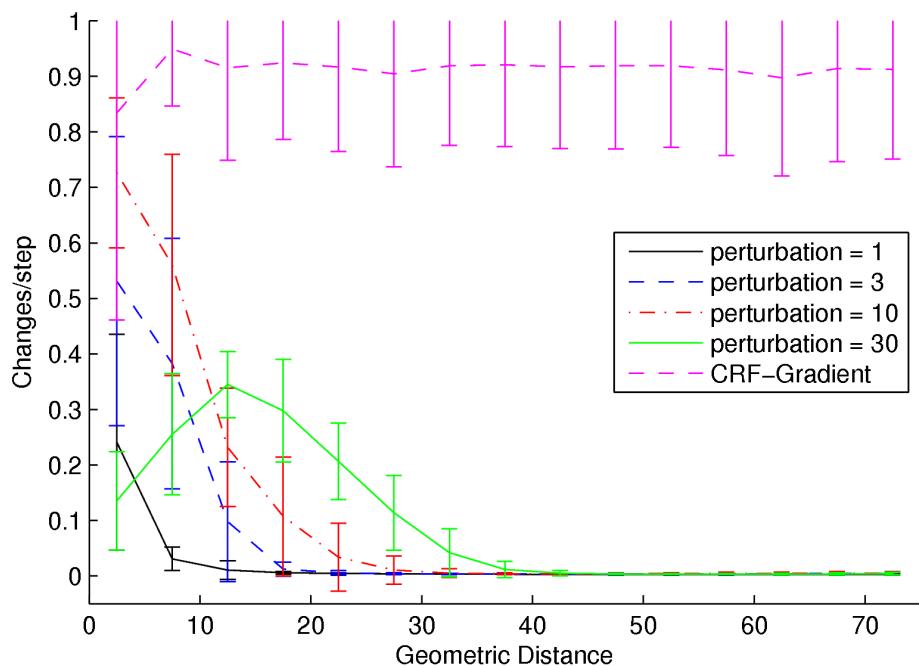
- Sources take $g_x(t)=0$
- Else measure maximum slope and minimum distance through neighbors (w. r/δ distortion):
 - If value is more than 2x lowest value through neighbor, snap to slope=1
 - Else if slope is not ε -acceptable, make ε -acceptable
 - Once every $g_x(t)$ updates, use $\varepsilon=0$

Flex-Gradient vs. CRF-Gradient

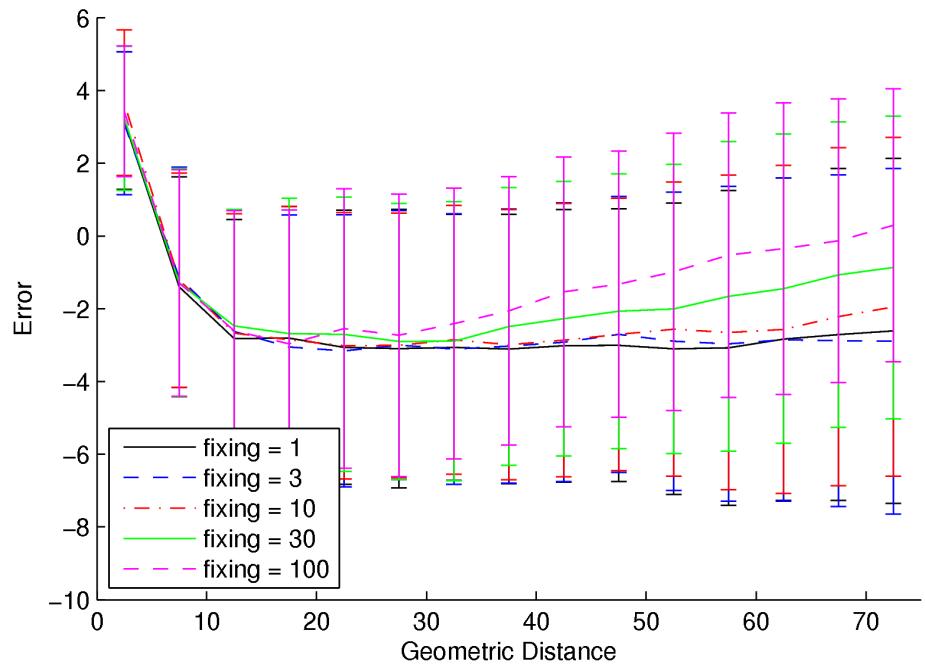
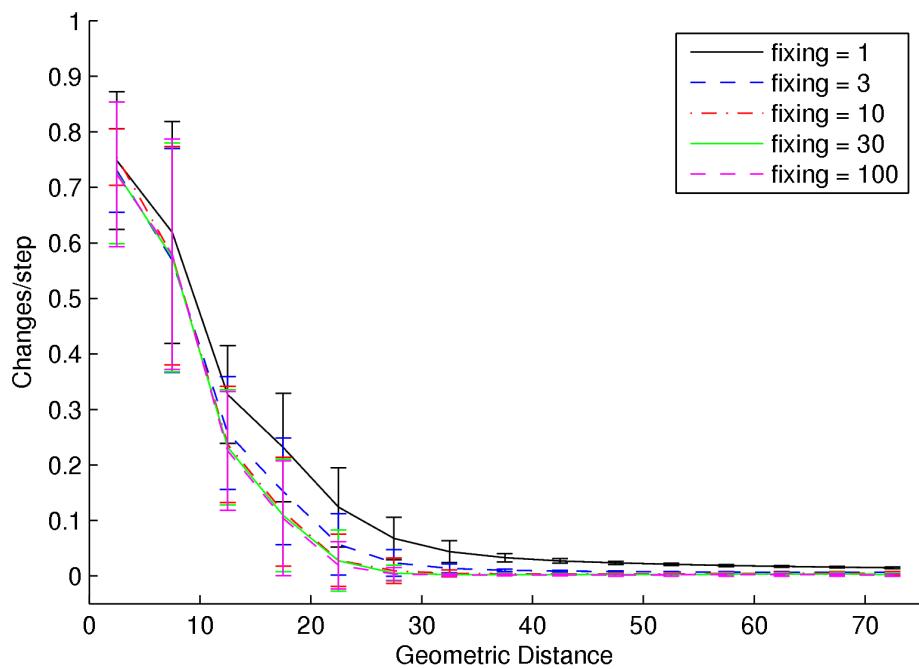


```
proto -n 1000 -r 10 -led-stacking 2 "(flex-gradient-demo 0.3 10 0.2 1 1)" -l -s 1 -w -m
```

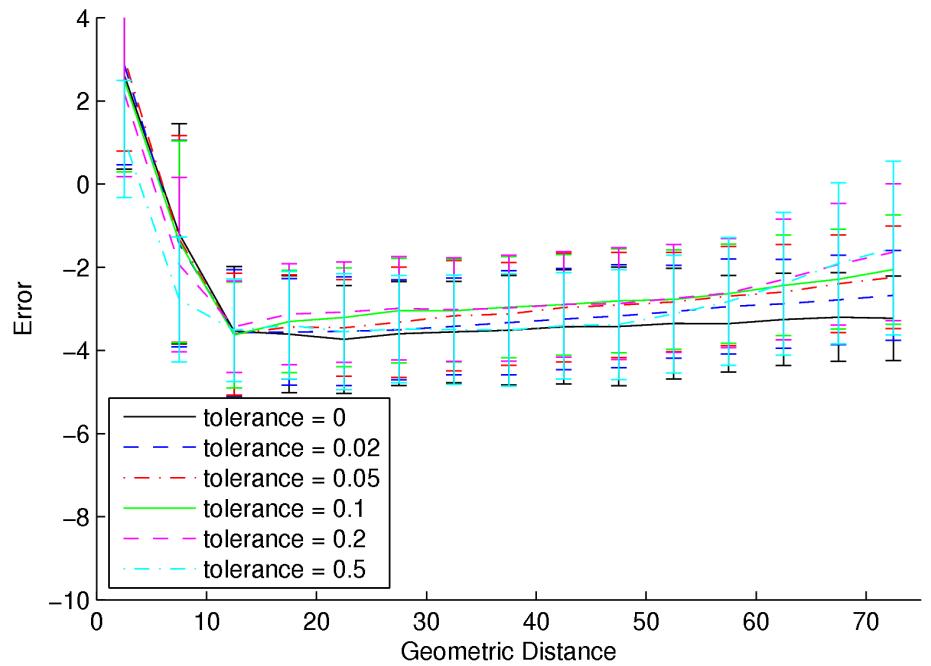
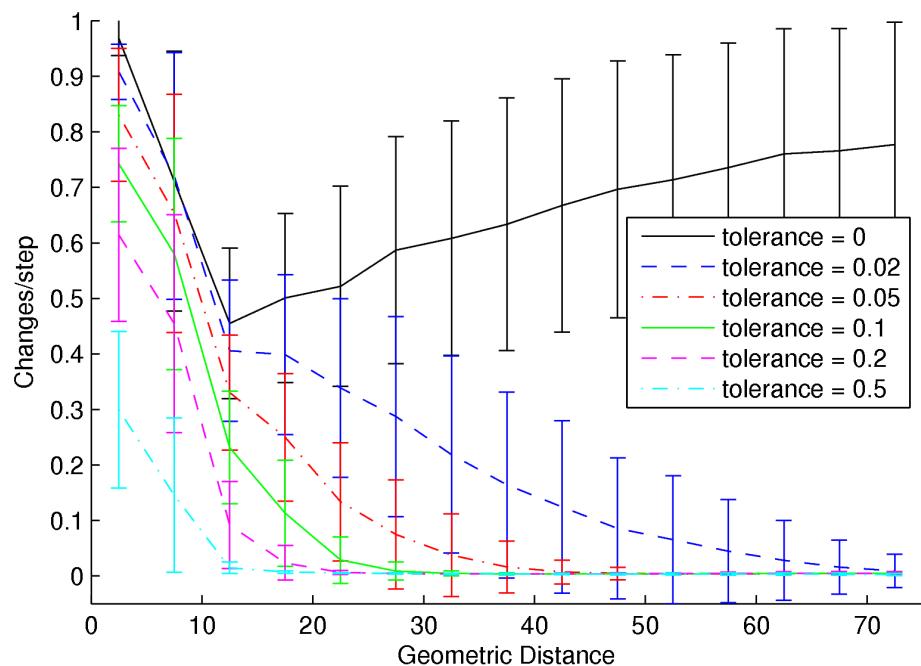
Perturbations affect limited range



Even infrequent repair helps



A little tolerance goes a long way



Contributions

- Tolerating small errors can reduce communication cost by orders of magnitude
- Flex-Gradient algorithm heals slope changes
 - Oscillation affects bounded radius
 - Long-term changes are propagated everywhere