

EDA (CS286.5b)

Day 7
Placement
(Simulated Annealing)

Assignment #1 due Friday

Today

- Placement
- Simulated Annealing
 - big hammer
 - iterative improvement
 - randomness
 - avoid local minima

Placement

- **Problem:** Pick locations for all building blocks
 - minimizing energy, delay, area
 - really:
 - minimize wire length
 - minimize channel density

Bad Placement

- How bad can it be?
 - Area
 - delay
 - energy

Bad: Area

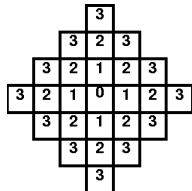
- All wires cross bisection
- $O(N^2)$ area
- good: $O(N)$

Bad: Delay

- All critical path wires cross chip
- Delay = $O(|\text{PATH}|^2 * L_{\text{side}})$
 - [and L_{side} as $O(N)$]
- good: $O(|\text{PATH}| * L_{\text{cell}})$
- compare 100ps gates to many nanoseconds

Distance

- Can we place everything close?



“Closeness”

- Try placing “everything” close

Manhattan Distance	Places	Transitive Fanin
1	4	4
2	8	16
3	12	64
i	i	i
n	$4n$	4^n



Problem Characteristics

- Familiar
 - NP Complete
 - local, greedy not work
 - greedy gets stuck in local minima

Simulated Annealing

- Physically motivated approach
- Physical world has similar problems
 - objections/atoms seeking minimum cost arrangement
 - at high temperature (energy) can move around
 - at low temperature, no free energy to move
 - cool quickly, freeze in defects (weak structure)
 - cool slowly, allow to find minimum cost

Simulated Annealing

- At high temperature can move around
 - not trapped to only make “improving” moves
 - free energy from temperature allows exploration of non-minimum states
 - avoid being trapped in local minima
- As temperature lowers
 - less energy to take big, non-minimizing moves
 - more local / greedy moves

Design Optimization

Components:

- “Energy” (Cost) function to minimize
 - represent **entire** state, drives system forward
- Moves
 - local rearrangement/transformation of solution
- Cooling schedule
 - initial temperature
 - temperature steps (sequence)
 - time at each temperature

Basic Algorithm Sketch

- Pick an initial solution
- Set temperature (T) to initial value
- while (T > Tmin)
 - for time at T
 - pick a move at random
 - compute Δcost
 - if less than zero, accept
 - else if $\text{RND} < e^{-\Delta\text{cost}/T}$, accept
 - update T

Details

- Initial Temperature
 - $T_0 = \Delta_{\text{avg}} / \ln(P_{\text{accept}})$
- Cooling schedule
 - fixed ratio: $T = \lambda T$
 - (e.g. $\lambda = 0.85$)
 - temperature dependent
- Time at each temperature
 - fixed number of moves?
 - Fixed number of rejected moves?
 - Fixed fraction of rejected moves?

Cost Function

- Can be very general
- Should drive entire solution in right direction
 - reward each good move
- Should be cheap to compute delta costs
 - e.g. FM

Cost Functions

- Total Wire Length
- Channel widths
 - probably wants to be more than just width
- Cut width

Bad Cost Functions

- Update cost
 - rerun maze route on every move
 - rerun timing analysis
 - recalculate critical path delay
- Drive toward solution:
 - size < threshold ?
 - Critical path delay

Initial Solution

- Spectral Placement (last time)
- Random
- Constructive Placement

Moves

- Swap two cells
- swap regions
 - ...rows, columns, subtrees
- rotate cell (when feasible)
- flip (mirror) cell
- permute cell inputs (equivalent inputs)

Variant

- Allow non-legal solutions
 - capture badness in cost function
 - *E.g.* -- allow cells to overlap
- Just make sure cost function makes very expensive as cool
 - settle out to legal solutions

Variant: “Rejectionless”

- Order moves by cost
 - compare FM
- Pick random number first
- Use random to define range of move costs will currently accept
- Pick randomly within this range
- (never pick a costly move which will reject)

Theory

- If stay long enough at each cooling stage
 - will achieve tight error bound
- If cool long enough
 - will find optimum

Practice

- Good results
 - ultimately, what most commercial tools use...
- Slow convergence
- Tricky to pick schedules to accelerate convergence

Big Hammer

- Costly, but general
- Works for most all problems
 - (part, placement, route, retime, schedule...)
- Can have hybrid/mixed cost functions
 - as long as weight to single potential
- With case, can attack multiple levels
 - place and route
- Ignores structure of problem
 - resignation to finding/understanding structure

Summary

- Placement
 - problem
 - importance
- Simulated Annealing
 - use randomness to explore space
 - accept “bad” moves to avoid local minima
 - decrease tolerance over time
- General purpose solution
 - costly in runtime

Today’s Big Ideas:

- Exploit “freedom” in problem to reduce costs
- Use randomness to explore large space