Maximizing the Sum of Radii of Disjoint Balls or Disks

David Eppstein

28th Canadian Conference on Computational Geometry (CCCG 2016)

Vancouver, Canada, August 2016
Tradeoff in label size for map labeling

Too small: hard to find among other features

Too big: overlap each other, difficult to separate

Depends on *local density* more than absolute size
Goal: Find maximum feasible label size

Formally: Place non-overlapping circles with given centers, maximizing some objective function. But what to maximize?

Max min radius: easy (min dist/2) but too global (one close pair makes all circles small)

Max total area: too unbalanced, leads to zero-radius circles

Max sum of radii: connected circles can stay balanced, disconnected circles vary independently
Detour through abstract metric spaces

Metric space: points with a symmetric non-negative distance function that obeys the triangle inequality: a shortest path from $x$ to $y$ is never longer than a path from $x$ to $y$ passing through $z$

Example: Any finite set of points in $\mathbb{R}^2$ and their distances
Wrong definition: Ball = \{ points within distance $r$ of center \}
Balls overlap when their intersection is nonempty

Difficult to use computationally
Changes when you embed the space into one with more points

Right definition: Ball = pair (center, radius)
Balls overlap when sum of radii $> \text{distance of centers}$
Given a finite metric space $(X, d)$ (the circle centers and their distances):

- Choose a radius $r_i \geq 0$ for each center $x_i$ in $X$
- Obey non-overlapping circle constraints $r_i + r_j \leq d(x_i, x_j)$
- Maximize $\sum r_i$

This is a linear program!

...but does it have a combinatorial solution?
Linear programming duality

Every linear program has a *dual* with:

- a variable for each primal constraint
- a constraint for each primal variable
- the same solution value

Our linear program’s dual is:

- Find a weight \( w_{ij} \geq 0 \) for each pair \((i, j)\)
- With each point \( x_i \) having total weight \( \sum_j w_{ij} \geq 1 \)
- Minimizing \( \sum_{i,j} w_{ij} d(x_i, x_j) \)

This is the LP relaxation of minimum-length perfect matching on the complete graph of the given center points

Matching: all weights \( w_{ij} \) are 0 or 1; matched edges have weight 1

Relaxation: optimal weights may be 0, 1, or 1/2
Choose $2w_{ij}$ edges between each pair of points $(x_i, x_j)$

The result is the minimum-length 2-regular multigraph over $K_n$
(a partition of the vertices into odd cycles and 2-cycles)

Equivalent (up to unimportant choice of orientation for >2-cycles)
to minimum-length matching of the bipartite double cover $K_2 \times K_n$, a graph with two vertices for each input point $x_i$
From matching back to optimal radii

Most bipartite matching algorithms are *primal-dual*, giving both matched edges and variables of the dual of the LP relaxation.

Applying this to matching on $K_2 \times K_n$ gives us two dual variables per vertex: radii of red and blue circles such that each red-blue pair with different centers are non-overlapping.

Averaging these two variables gives one optimal radius per center.
A better graph than the complete graph

We need a supergraph of the optimal 2-regular multigraph
...but it doesn’t need to be the complete graph

Instead, use intersection graph of balls with radius = nearest neighbor distance
Properties of nearest-neighbor intersection graph

- Smallest disk intersects $O(1)$ others
- $\#\text{edges} = O(n)$
- *Separator theorem*: split into constant-factor-smaller pieces by removing $O(n^{1-1/d})$ disks
- Can be constructed in time $O(n \log n)$ (for constant $d$)
Separator-based weighted bipartite matching

- Construct separator hierarchy
- With separator hierarchy already constructed, shortest paths take linear time [Henzinger et al., JCSS 1997]
- Recursively solve weighted matching for two subgraphs whose intersection is separator and whose union is the whole graph
- For each separator vertex, set dual variable to min from two subproblems and keep matched edge from that subproblem
- Use fast shortest path algorithm to find augmenting paths ($\leq 1$ per separator vertex) until no more can be found

$$\text{Time} = \text{separator size} \times O(n)$$

Shaves a log from best published bound by Lipton & Tarjan (1980)
Computes dual variables, not just the matching itself
Putting it all together

Weighted matching on $K_2 \times$ nearest-neighbor intersection graph

Average two dual variables per point to get optimal radii

Time $O(n^3)$ in metric spaces, $O(n^{2-1/d})$ in Euclidean spaces

Optimal solution = odd cycles + pairs of tangent disks