

An Optimal SDP Algorithm for Max Cut and an Equally Optimal Long Code Test

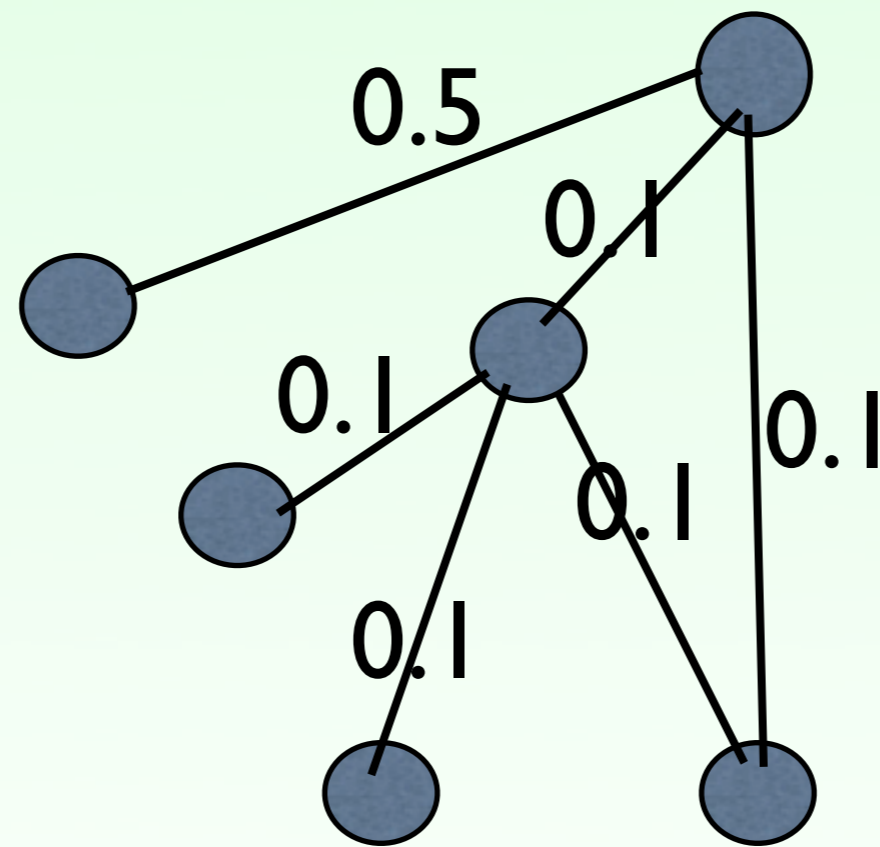
Yi Wu

yiwu@cs.cmu.edu

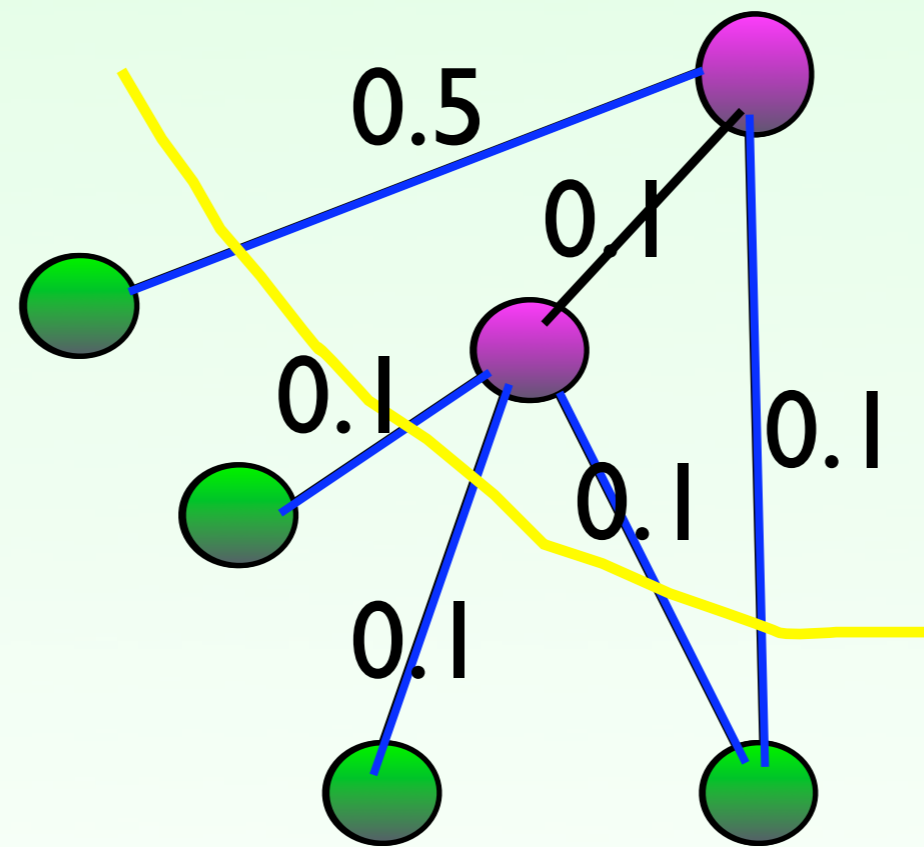
Carnegie Mellon University

joint work with Ryan O'Donnell

The Max-Cut Problem



The Max-Cut Problem



Quadratic Form and SDP Relaxation

Input is graph G with edge weight sums to 1:

$$\sum_{i,j} w_{ij} = 1$$

QP : Find $x_i \in \{-1, 1\}, i = 1..n$ that maximizes

$$opt(G) = \max \sum_{i,j} \frac{1 - x_i x_j}{2} w_{ij}$$

SDP: Find $v_i \in \mathbf{R}^n, |v_i| = 1, i = 1..n$ that maximizes

$$sdp(G) = \max \sum_{i,j} \frac{1 - v_i v_j}{2} w_{ij}$$

GW Alg for Max Cut

Given is a max-cut instance

- Solve its **SDP form** and get v_1, v_2, \dots, v_n
- Choose a **random gaussian vector** r
- Output $x_i = \text{sgn}(r \cdot v_i)$

SDP-Approximation and Integrality Gap

- Three terms defined: $\text{sdp}(G) \geq \text{opt}(G) \geq \text{alg}(G)$
- Measure of SDP-Approximation Algorithm

$$\min_G \frac{\text{alg}(G)}{\text{sdp}(G)}$$

- The integrality gap

$$\min_G \frac{\text{opt}(G)}{\text{sdp}(G)}$$

- GW is a **0.878** SDP algorithm and the integrality gap is also **0.878**. (Kar 99, FS02)

Problem of GW

- For Max-Cut with optimum cut very small (close to 1/2 and so-called **light max-cut**), GW performs even worse than **random assignment**:

[ASZ01]: There is G such that:

$$sdp(G) = 0.51$$

$$alg_{GW}(G) = 0.51 * 0.878 \quad alg_{random}(G) = 0.5$$

Redefine the Problem

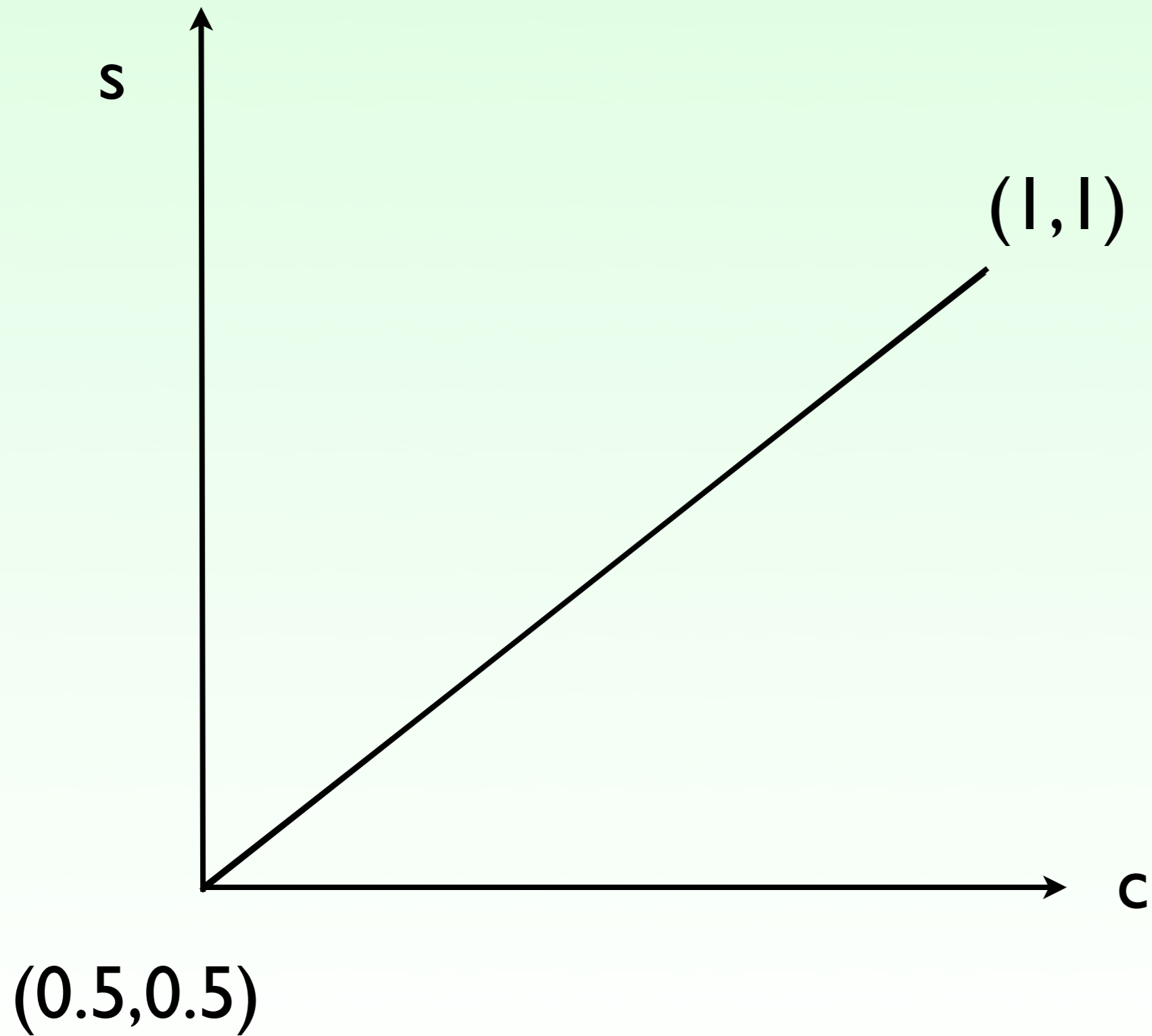
- Integrality Gap

$$Gap(c) = \min_{sdp(G)=c} \frac{opt(G)}{sdp(G)}$$

- SDP Approximation

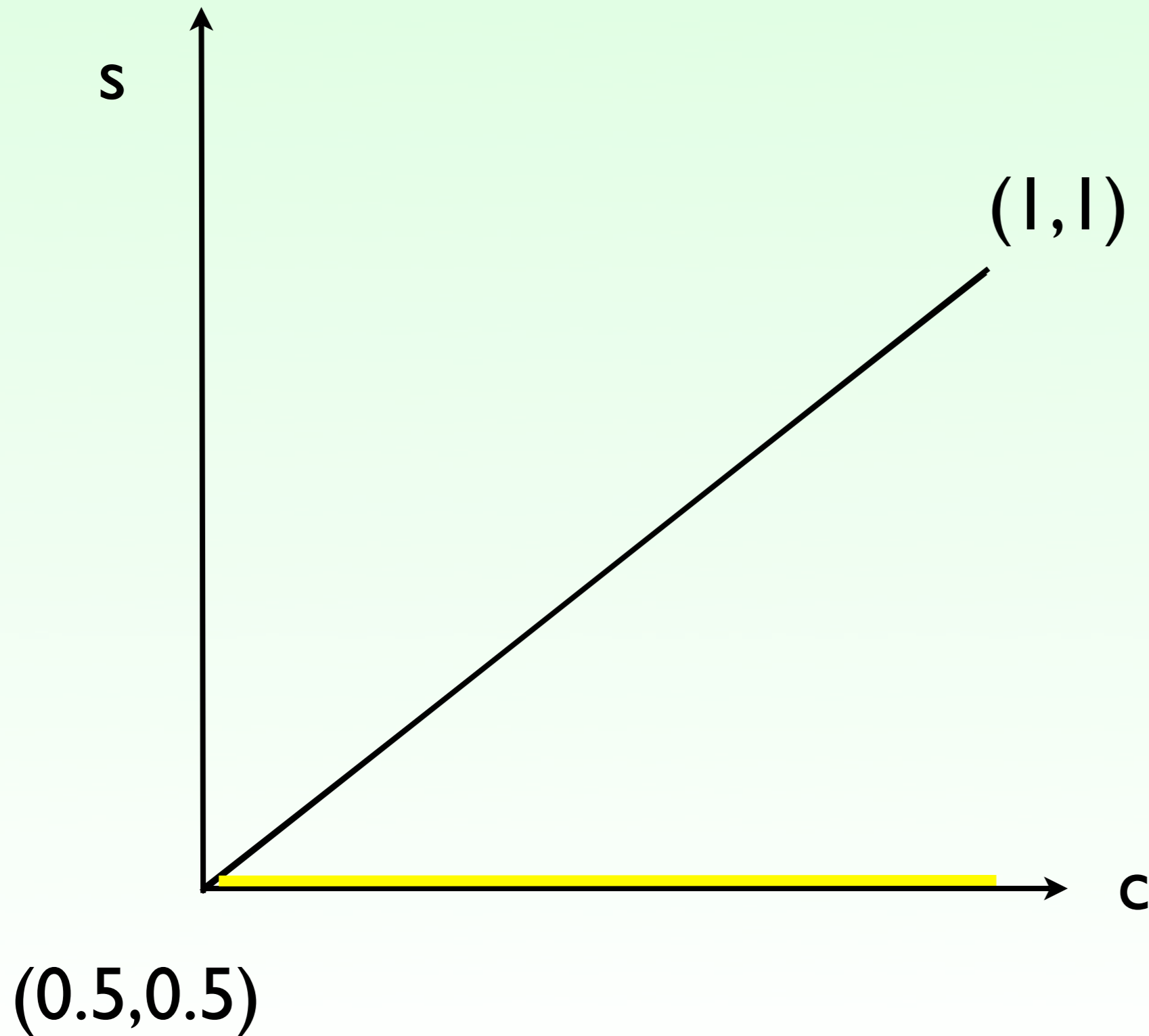
$$alg(c) = \min_{sdp(G)=c} \frac{alg(G)}{sdp(G)}$$

Previous Result



Previous Result

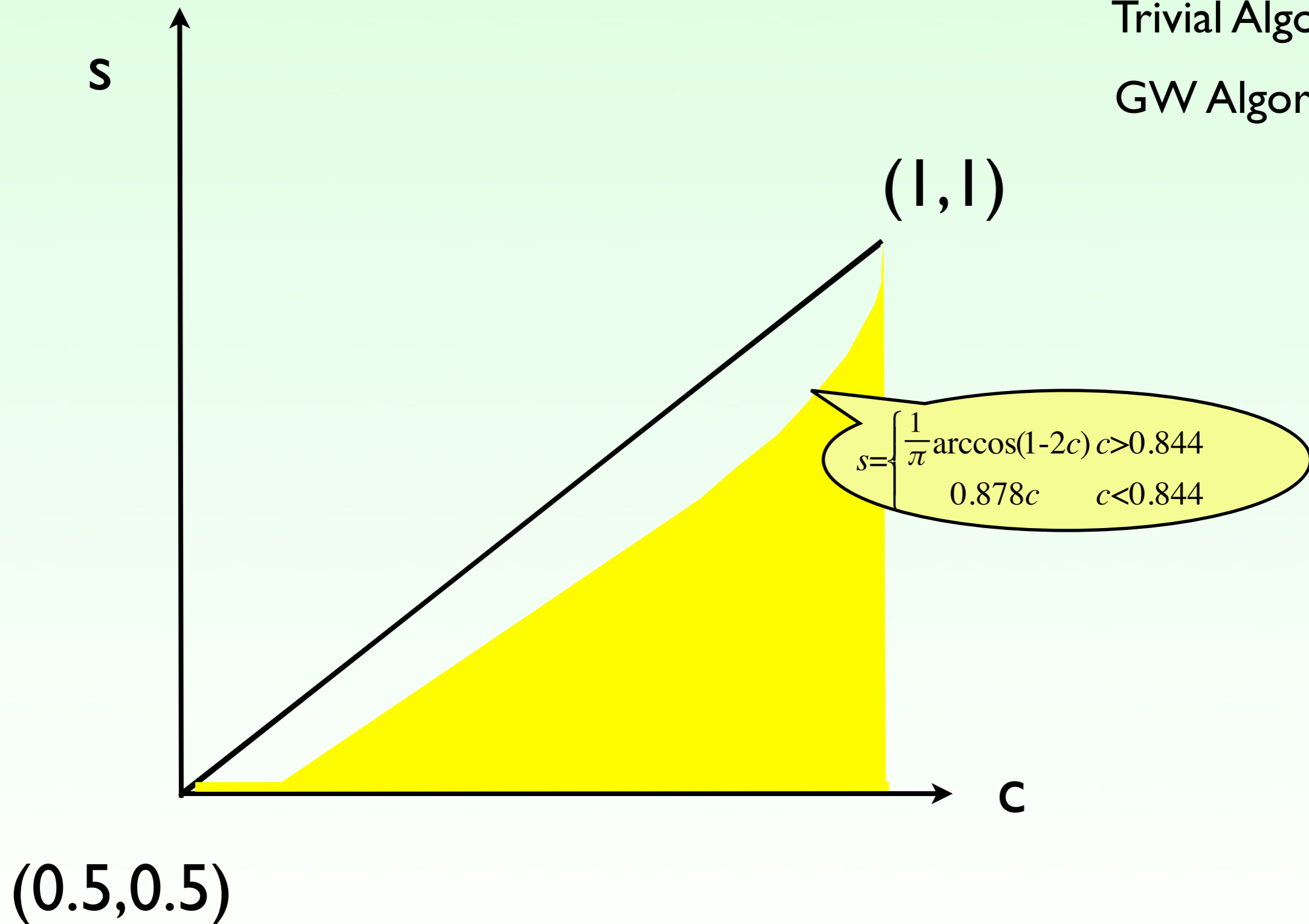
Trivial Algorithm



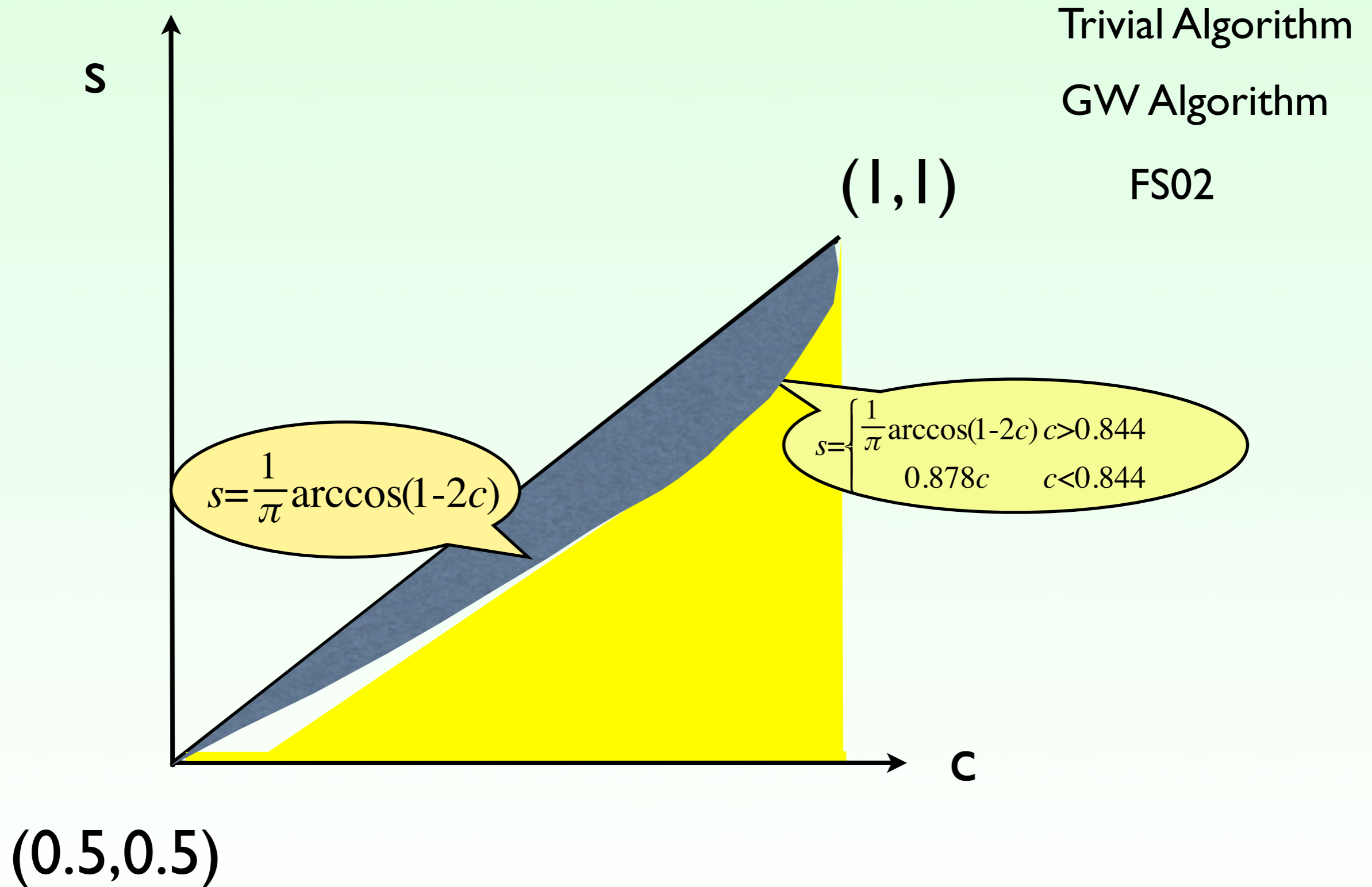
Previous Result

Trivial Algorithm

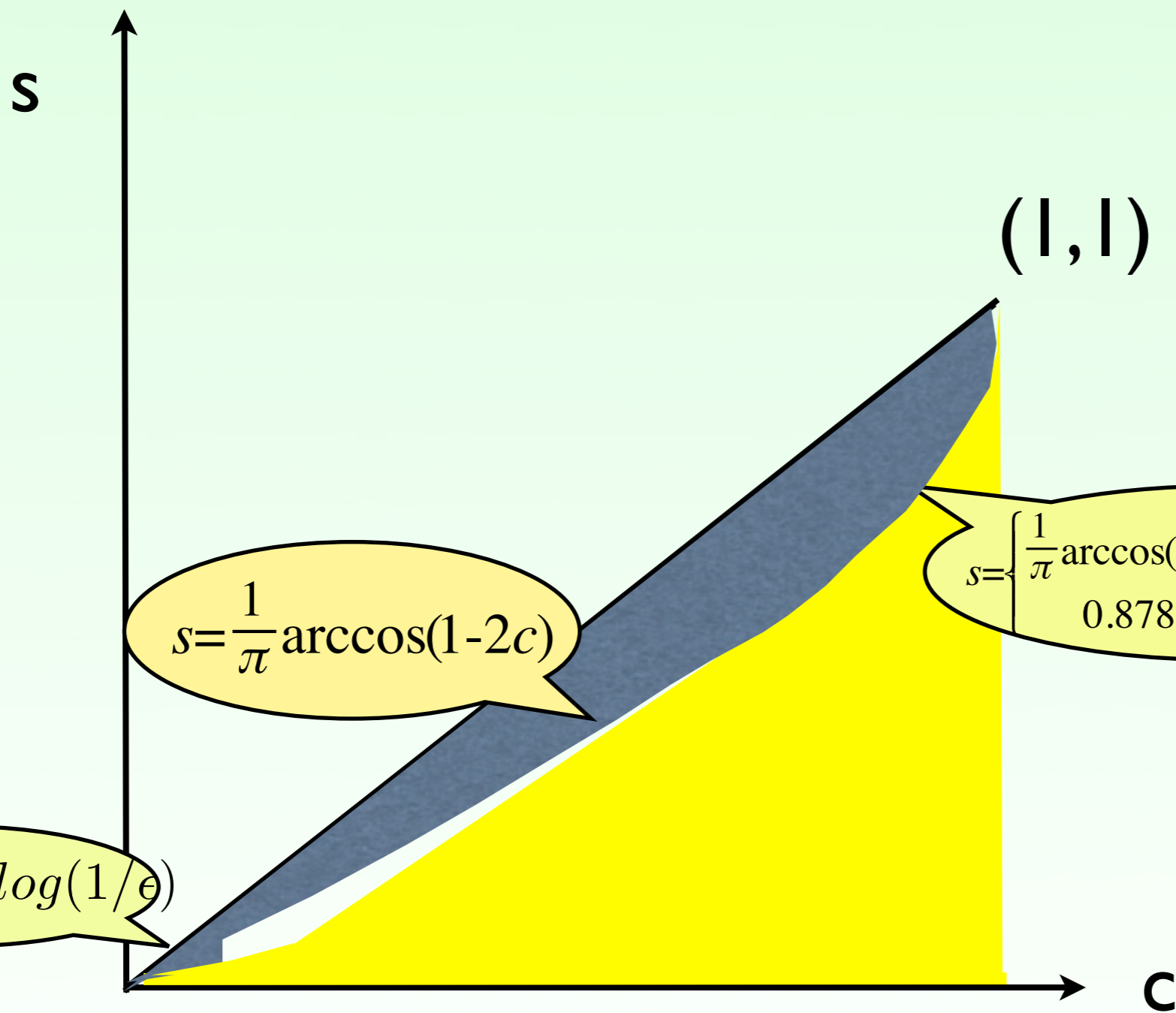
GW Algorithm



Previous Result



Previous Result



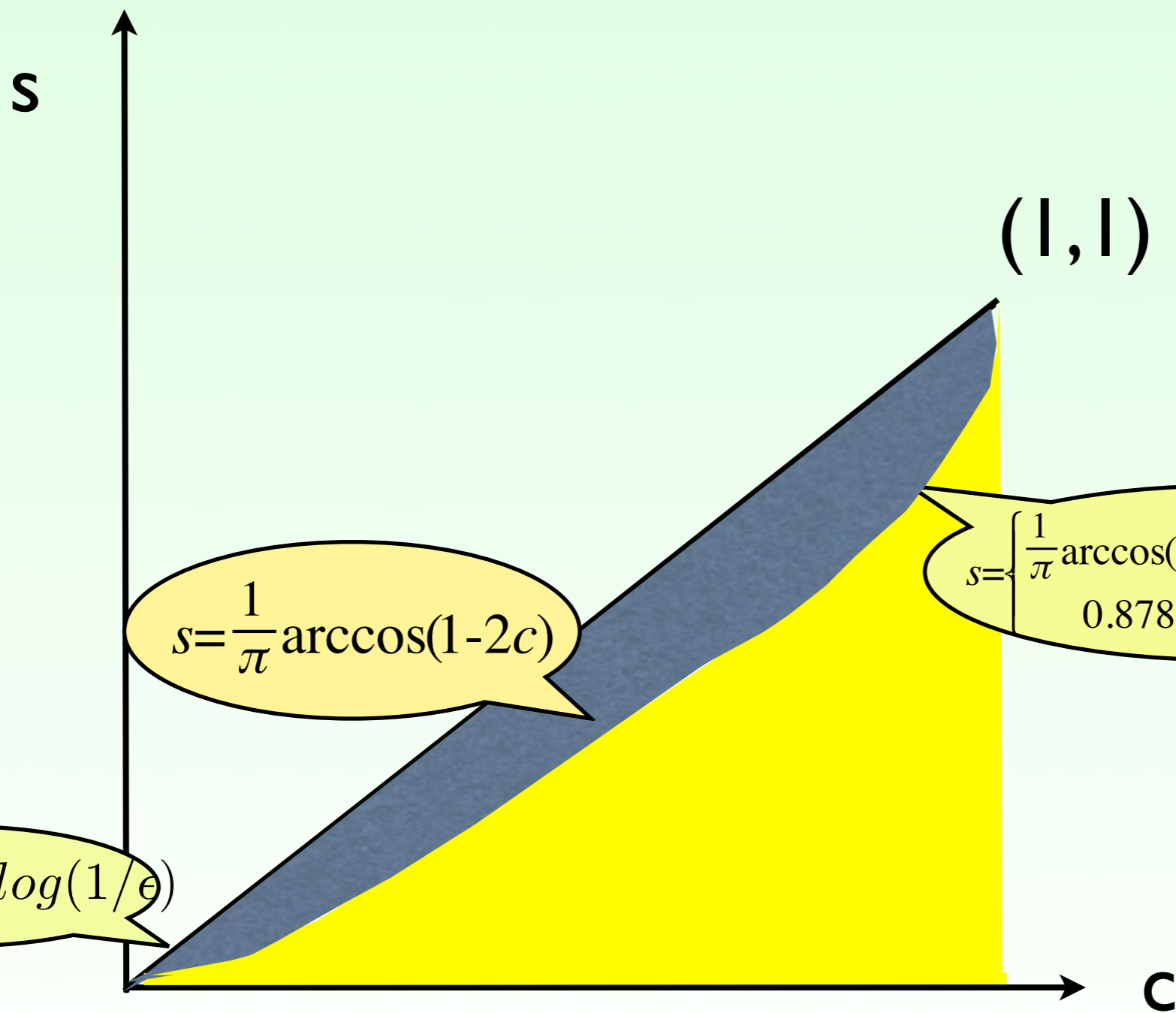
Trivial Algorithm

GW Algorithm

FS02

Zwick99, FL01, CW04,
KO06

Previous Result



Trivial Algorithm

GW Algorithm

FS02

Zwick99, FL01, CW04,
KO06

Our Result

- ❑ Decide the **integrality Gap** for every c in polynomial time.
- ❑ Give an **polynomial** Algorithm achieving SDP-Approximation equal to the **integrality gap** for every c .
- ❑ Prove NP-Hardness of getting a better (Opt)-approximation algorithm assuming Unique Game Conjecture

Our Result

- Decide the **integrality Gap** for every c in polynomial time.
- Give an **polynomial** Algorithm achieving SDP-Approximation equal to the **integrality gap** for every c .
- Prove NP-Hardness of getting a better (Opt)-approximation algorithm assuming Unique Game Conjecture

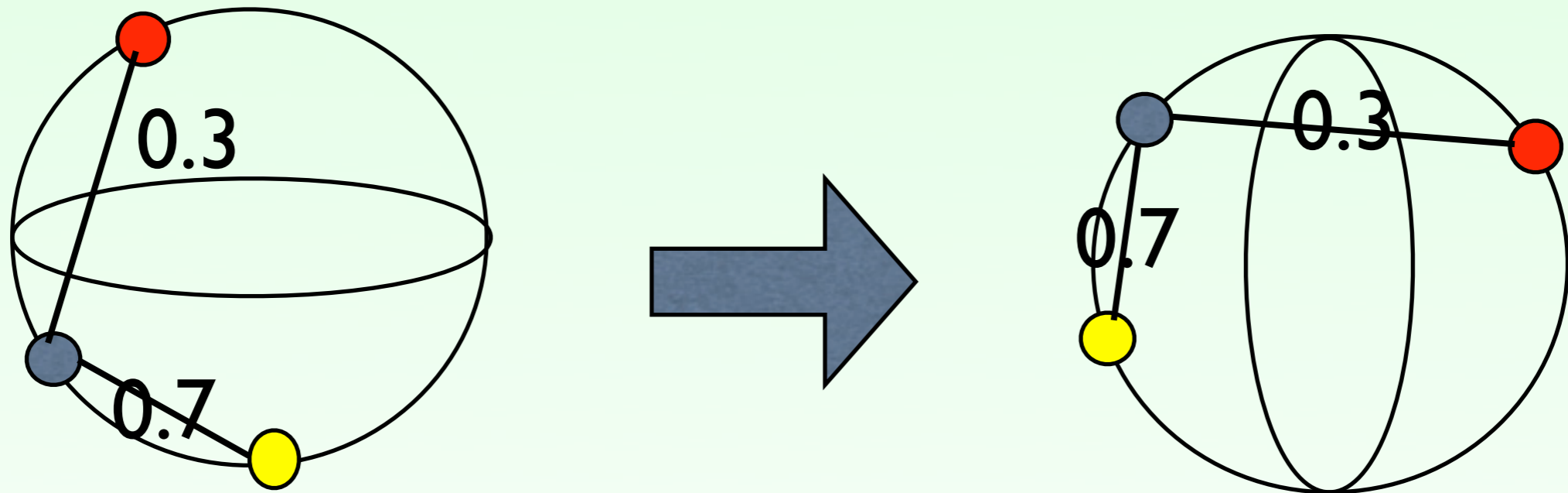
Understanding the worst graph

Given $\text{sdp}(G) = c$, we want to understand which graph has the minimum $\text{opt}(G)$?
Or which embedding G with sdp value c has smallest $\text{opt}(G)$?

Operation that keeps $\text{opt}(G)$ small

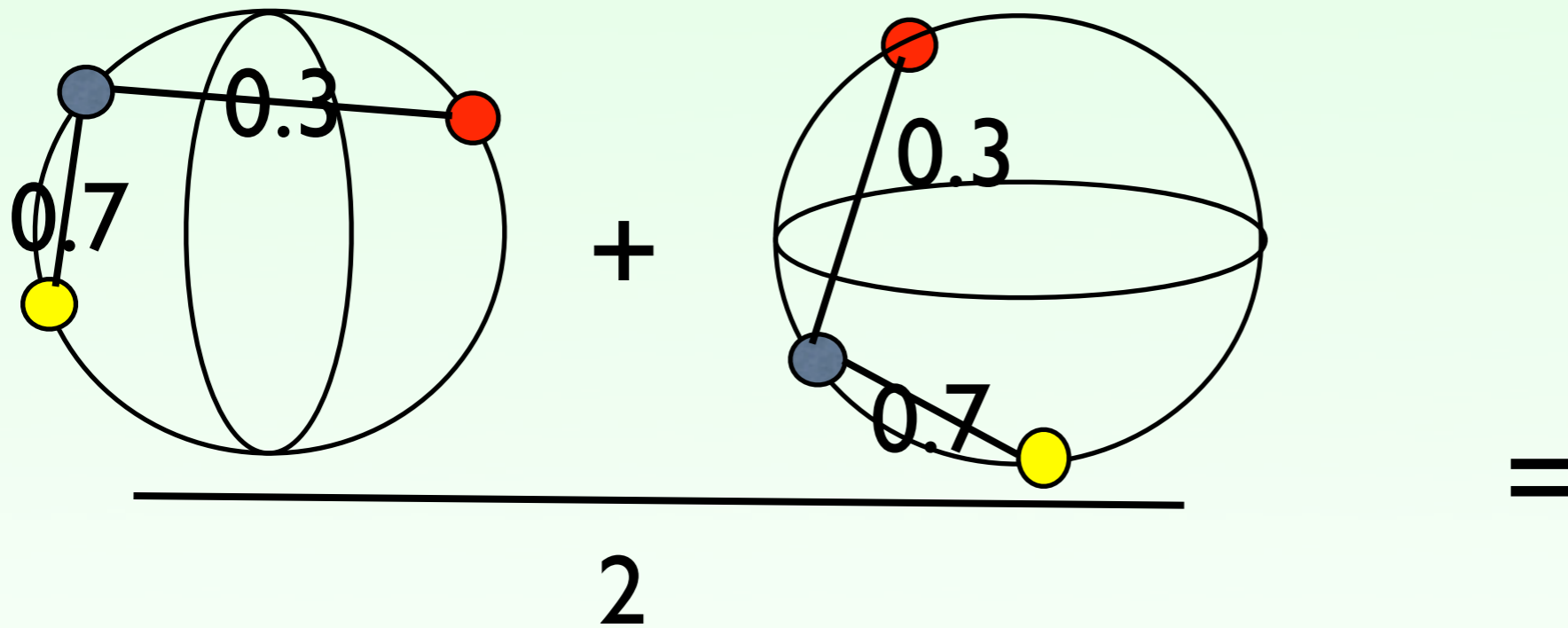
- Rotation
- Averaging

Rotation

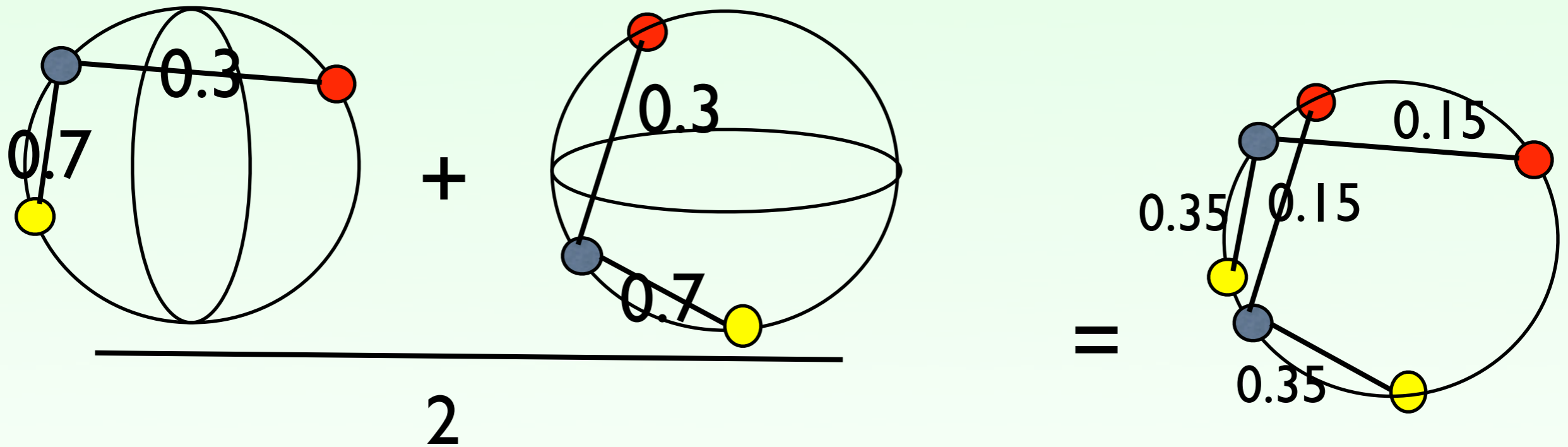


Rotation does not change pairwise inner product.

Averaging



Averaging



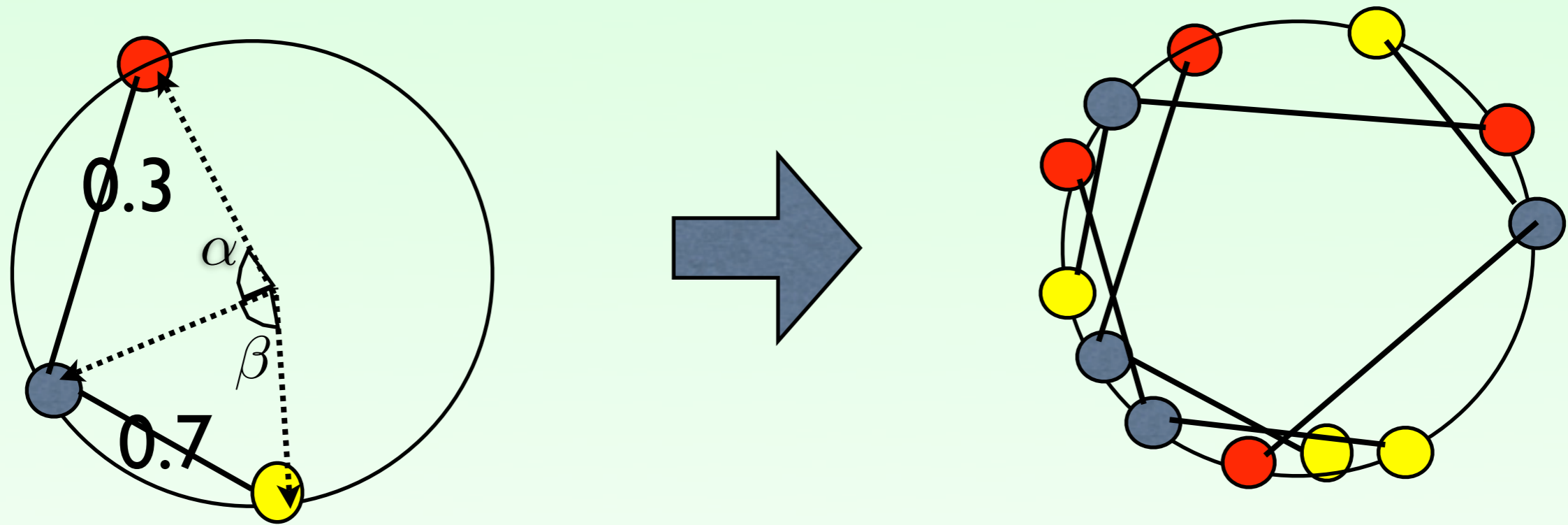
Symmetrization

Symmetrization of Graph G

1. Generate all possible rotation of G .
2. Take average of all possible rotations.

Finally we get symmetrized graph that still keeps the integrality gap.

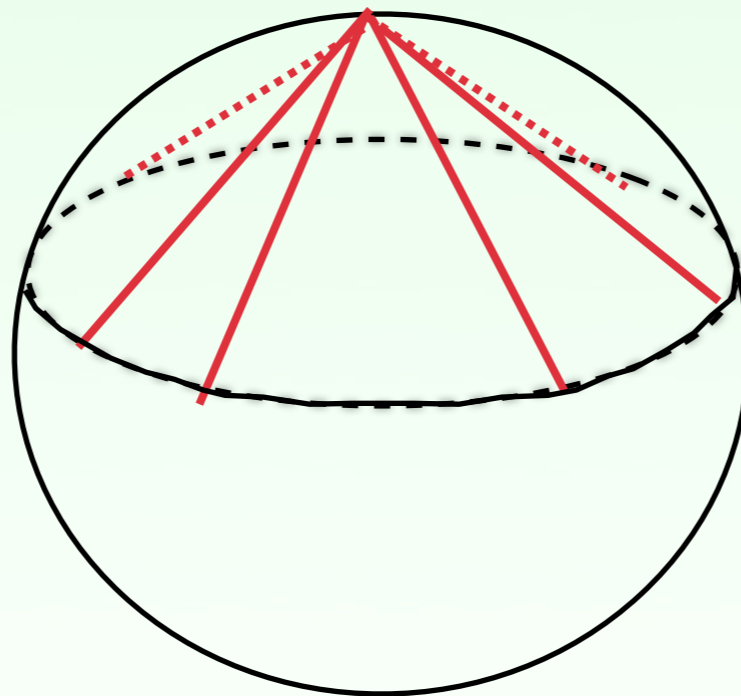
The Symmetrized graph



- In the limit, the symmetrized graph have:
- weight 0.3 is distributed uniformly to every two points with angle α .
 - weight 0.7 is distributed uniformly to every two points with angle β .

The Symmetrized graph

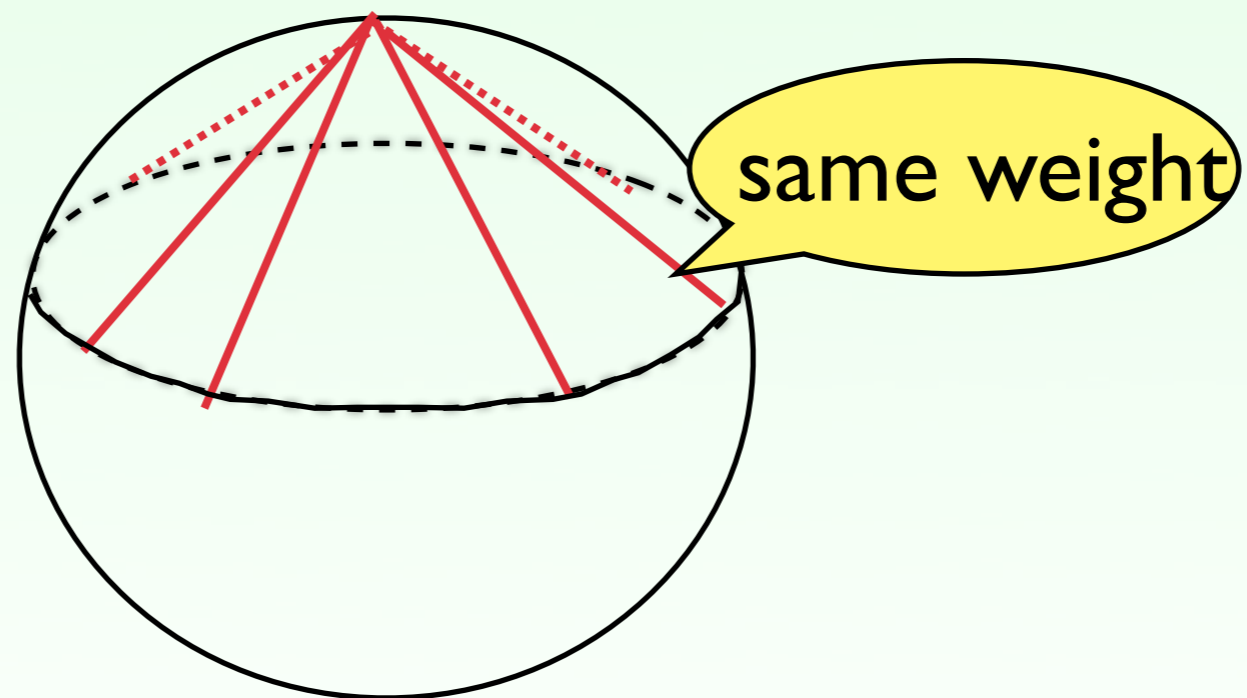
In general symmetrized graph, the weight on any two points is only determined by their inner product.



A graph is defined by a distribution on possible inner product value.

The Symmetrized graph

In general symmetrized graph, the weight on any two points is only determined by their inner product.



A graph is defined by a distribution on possible inner product value.

Remaining Proof

- The integrality gap instance is a (ϵ, ρ) symmetrized graph. (Convexity)
- The symmetrized graph's optimal rounding function is **one dimensional, monotone and odd**. (Hermite Analysis, Minmax Theorem)
- Formulate calculating integrality gap as a convex optimization problem. (KKT)

Future Problem

- Understanding stronger SDP
- Understanding other CSP problem