Simple approximation algorithm for nonoverlapping local alignments

Piotr Berman¶
Department of Computer Science
Pennsylvania State University
University Park, PA 16802
Email: berman@cse.psu.edu

Bhaskar DasGupta‡
Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60607
Email: dasgupta@cs.uic.edu

S. Muthukrishnan
AT&T Labs – Research
180 Park Avenue
Florham Park, NJ 07932
Email: muthu@research.att.com

¶ Supported by NSF grant CCR-9700053, NLM grant LM05110 and DFG grant Bo 56/157-1.

‡ Supported by NSF Grant CCR-9800086.

Overview of this presentation

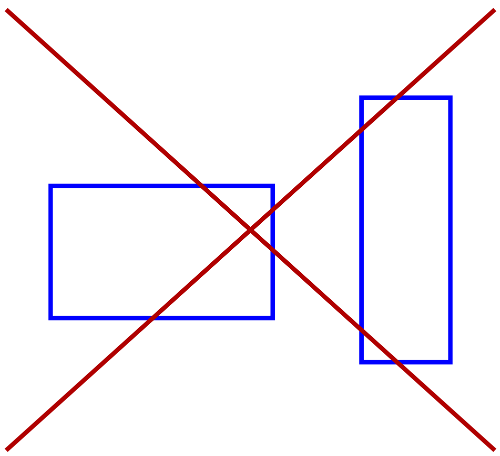
Nonoverlapping local alignments via rectangles

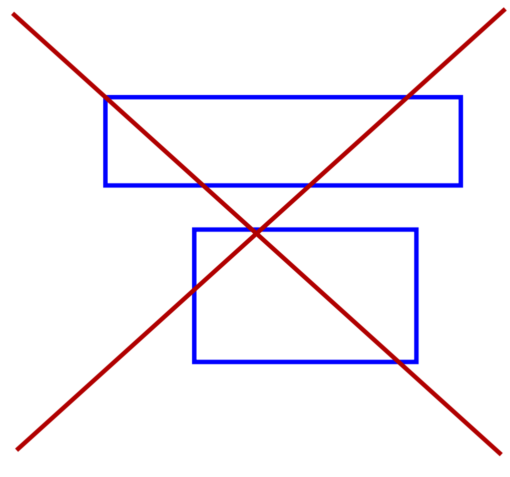
- Previous work
- Our results

Future research topics

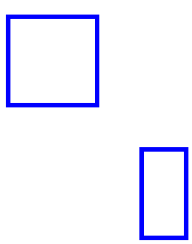
The Problem

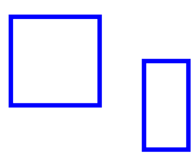
Given a set of weighted axis-parellel rectangles such that projections of no two rectangles enclose each other on the the x or y axes

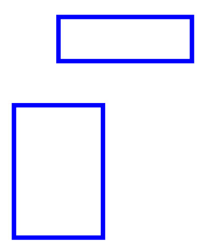




A pair of rectangles is independent if their projections on both axes are disjoint





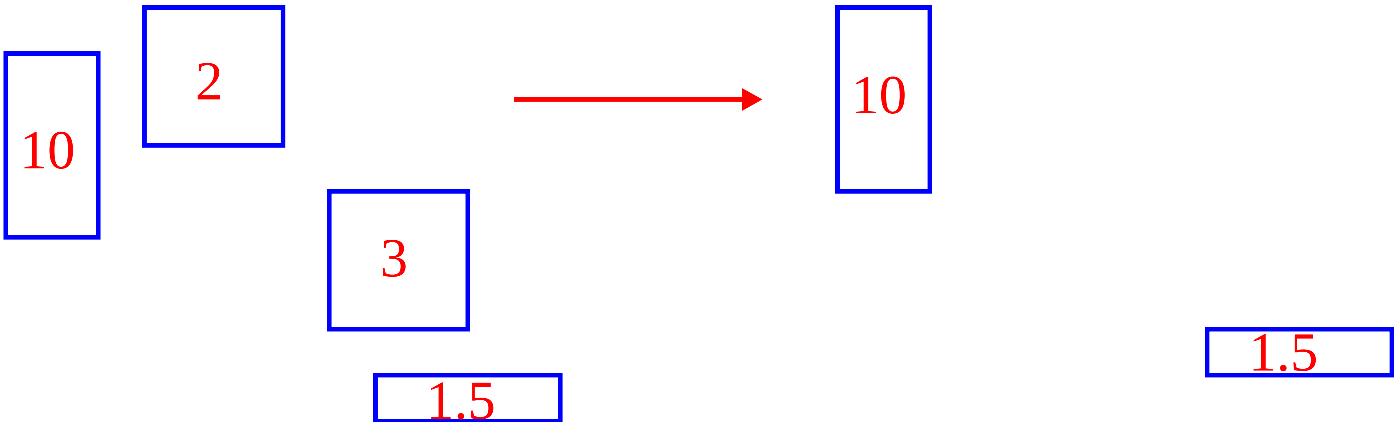


not independent   not independent

independent

Goal: Find a maximum-weight independent subset of rectangles

Example



input

an optimal solution of total weight 11.5

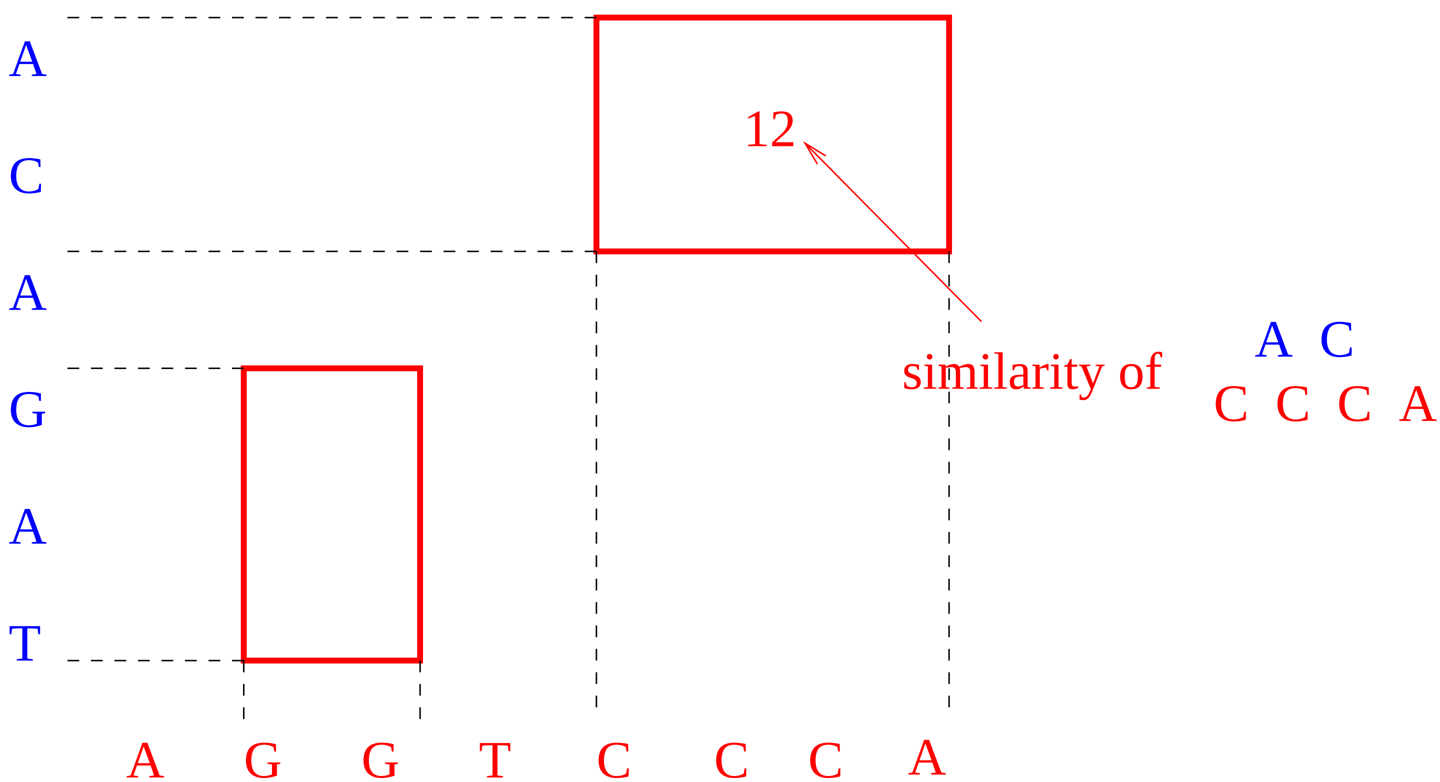
Biological motivation

Selection of fragments of high local similarity between two strings (between d strings for this problem in d dimensions)

Useful for studies on distances between sequences based on genome rearrangements

Biological motivation

Finding regions of local similarities in two sequences



An approximation algorithms for a maximization problem has a performance ratio (or approximation ratio) of r if

$$\begin{array}{c}\text{value of objective function}\\ \text{computed by algorithm}\end{array} \geqslant \frac{1}{r}\left(\begin{array}{c}\text{maximum value}\\ \text{of objective function}\end{array}\right)$$

Previous results

Bafna, Narayanan and Ravi (WADS'95)

- NP-complete
- Approximation algorithm with performance ratio 3.25
  - Converts to a problem of finding maximum-weight independent set in a 5-clawfree graph
  - Gives approximation algorithm for d+1-clawfree graphs with performance ratio $d-1+\frac{1}{d}$

Halldorsson (SODA'95)

- Approximation algorithm with performance ratio of about 2.5 when all weights are 1
  - Gives approximation algorithm for d+1-clawfree graphs with performance ratio of about $\frac{d+1}{2}$ when all weights are 1

Previous results (continued)

Berman (SWAT'00)

- approximation algorithm with performance ratio $2.5 + \varepsilon$

  taking at least $\Omega(n^4)$ time

  finds a $\frac{d+1}{2}$ - approximation of a (d+1)-clawfree graph

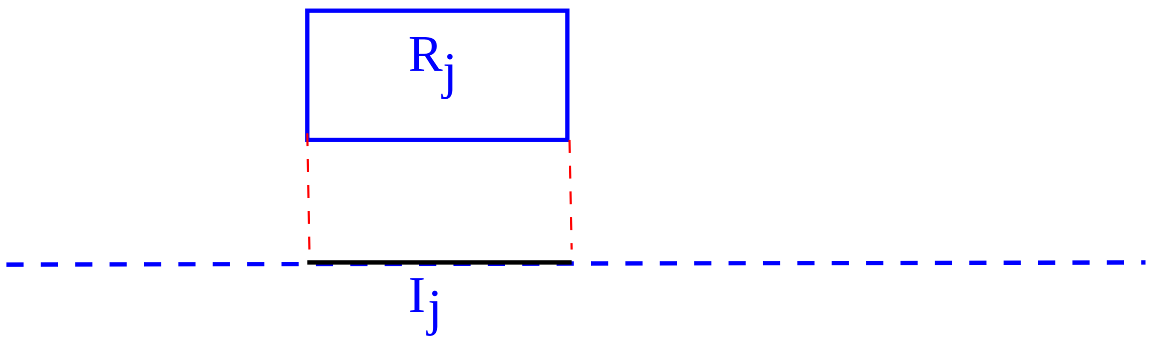
Our results
(n is the number of rectangles)

- Approximation algorithm with performance ratio 3 runs in $O(n \log n)$ time
- In d dimension, the performance ratio is 2^d-1 runs in $O(n\, d \log n)$ time

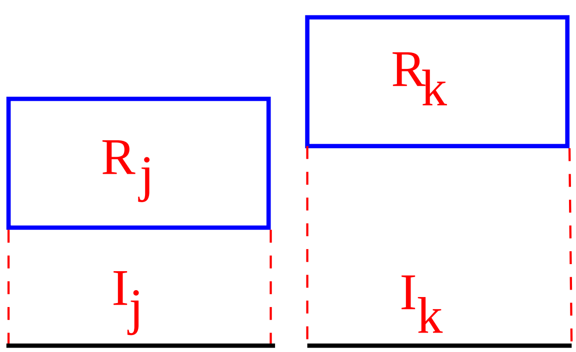
We use the two-phase technique of Berman and DasGupta (STOC'00)

Idea behind our algorithm

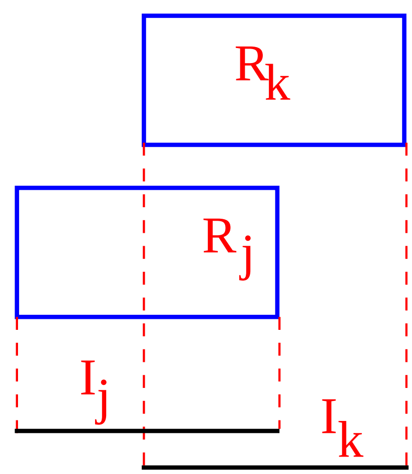
(a) Project each rectangle R_j as interval I_j on the x axis



Two intervals I_j and I_k conflict if their corresponding rectangles R_j and R_k are not independent



I_j and I_k conflict



I_j and I_k conflict

(b) Apply Two-phase algorithm, appropriately modified

Start with an initially empty stack S

First Phase (Evaluation Phase):

- Look at intervals in *non-decreasing* order of endings
- Evaluate a score v for each interval I_j (depends on scores of intervals in S and the weight of I_j)
- If $v > 0$, push I_j to S with score v

Second Phase (Selection Phase):

- pop the intervals in stack S one after another
- if appropriate, add the rectangle corresponding to this interval to our solution

Let $w(I)$ denote the weight of interval I

More details of evaluation phase

(evaluation of scores)

score v of an interval I_j is

$$w(I_j) - \sum_{I_k \in S; I_k \text{ conflicts with } I_j} w(I_k)$$

More details of selection phase

while (S is not empty)
{
  I = pop(S)
  if I does not conflict with already selected intervals, then insert I to our solution
}

For any interval I selected by the algorithm, let

$$b = \begin{cases} \textbf{maximum} \text{ number of intervals in any optimal} \\ \text{solution that have right endpoint later than } I \end{cases}$$

Theorem Our algorithm has a performance ratio of b

Idea of Proof: The proof proceeds in two stages.

(a) Consider end of evaluation phase

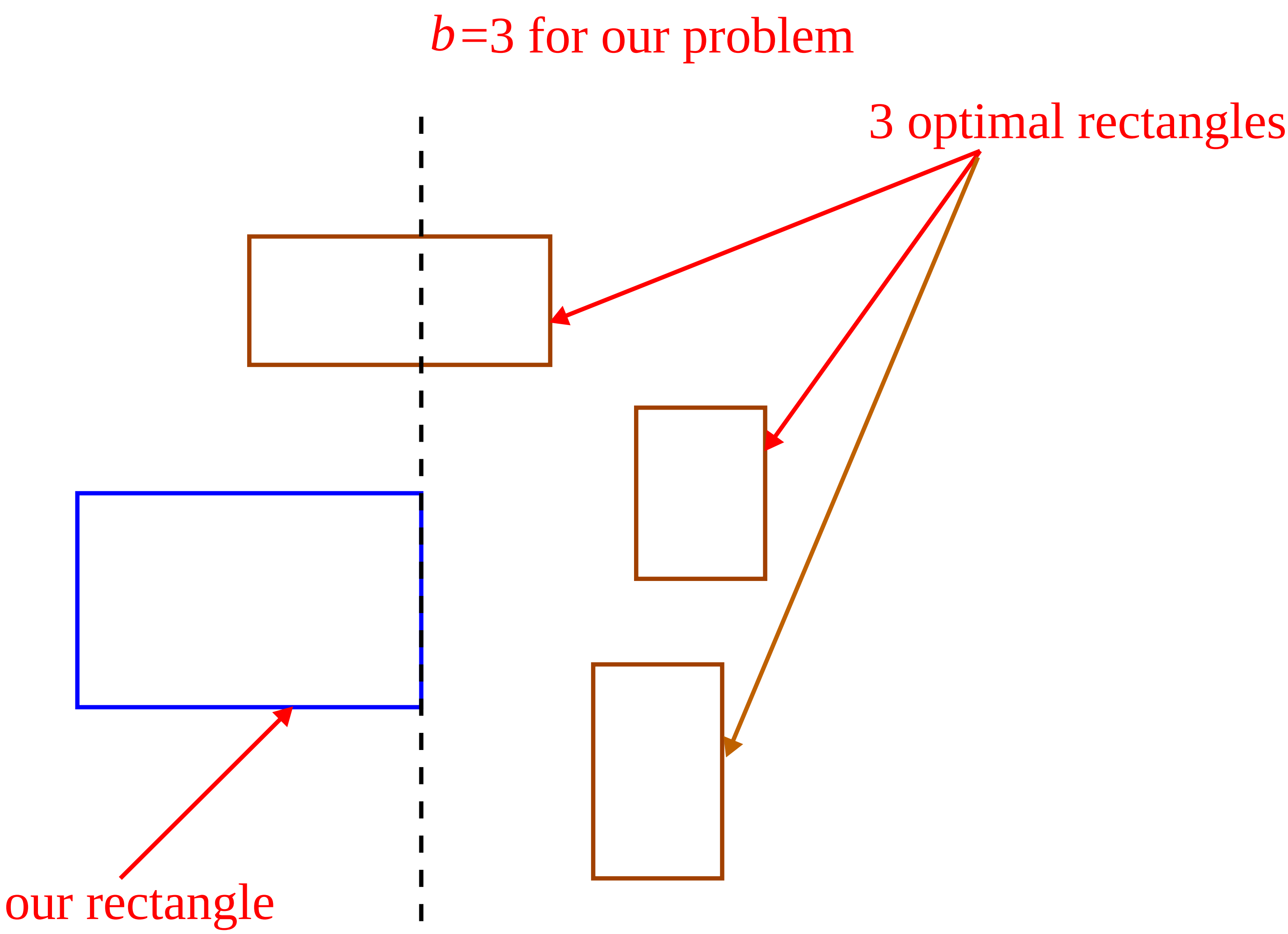
$$\begin{array}{lcl} V(S) & = & \text{sum of scores of intervals in stack S} \\ P & = & \text{total weight of an optimal solution} \end{array}$$

$$(\star) \quad \boxed{V(S) \geq \frac{1}{b}P}$$

(b) Consider end of selection phase

$$V = \text{total weight of our solution}$$

$$(\star\star) \quad \boxed{V \geq V(S)}$$



In d dimensions, $b \leqslant 2^{d}-1$

Future research topics

- Improved approximation algorithms

- Implement and test performance in actual applications

- Consider more complex objects than rectangles

- Add more meaningful biological constraints to the problem

  etc.