Protein Threading

BMI/CS 776
www.biostat.wisc.edu/bmi776/
Colin Dewey
cdewey@biostat.wisc.edu
Spring 2015

Goals for Lecture

the key concepts to understand are the following

- the threading prediction task
- the threading search task
- template models
- branch and bound search for threading

Protein Threading

- generalization of homology modeling
  - homology modeling: align sequence to sequence
  - threading: align sequence to *structure* (templates)

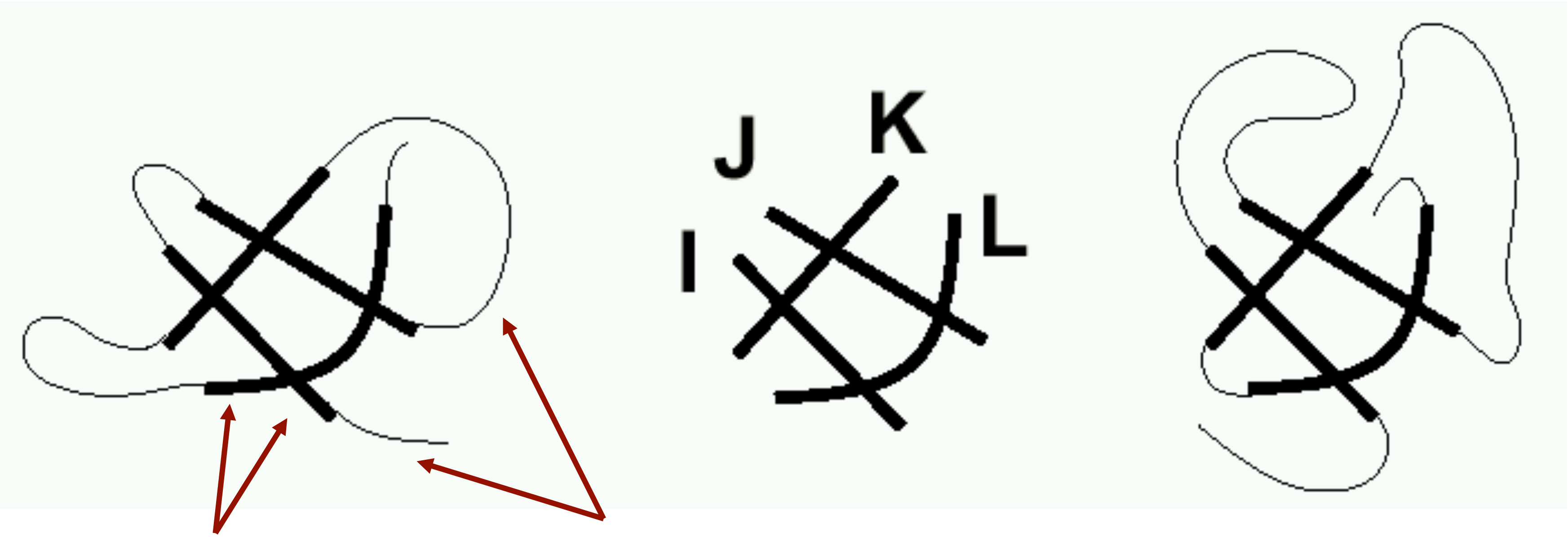
- key ideas
  - limited number of basic folds found in nature
  - amino acid preferences for different structural environments provide sufficient information to choose among folds

A Core Template

protein *A* threaded on template

template

protein *B* threaded on template



core secondary structure segments

loops

Figure from R. Lathrop et al. Analysis and Algorithms for Protein Sequence-Structure Alignment, in *Computational Methods in Molecular Biology*, Salzberg et al. editors, 1998.

Components of a Threading Approach

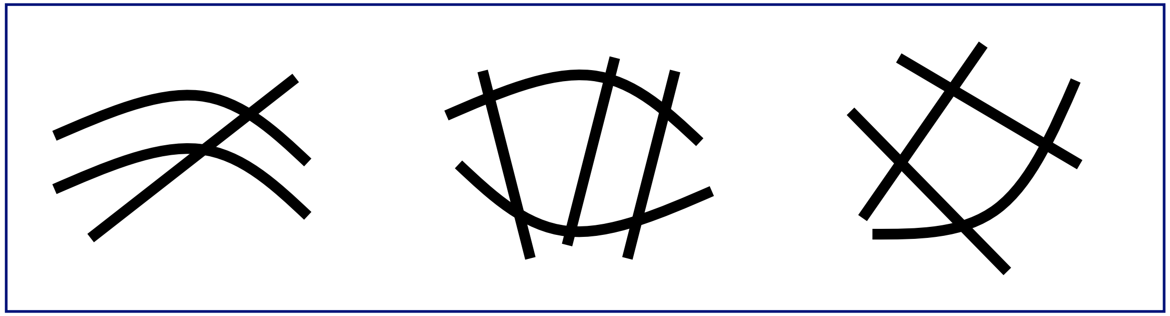
- library of core fold templates
- objective function to evaluate any particular placement of a sequence in a core template

✓ method for searching over space of alignments between sequence and each core template

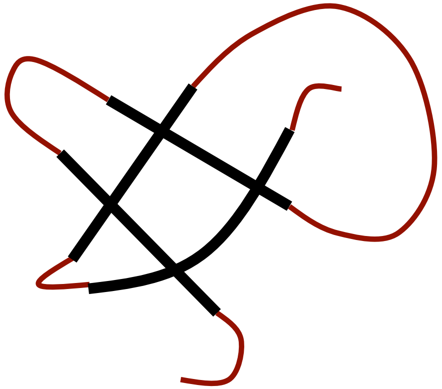
- method for choosing the best template given alignments

Task Definition: Prediction Via Threading

- given:
  - a protein sequence
  - a library of core templates



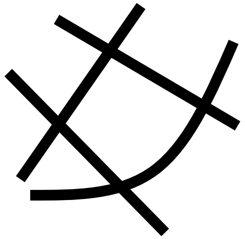
- return: the best alignment of the sequence to a template



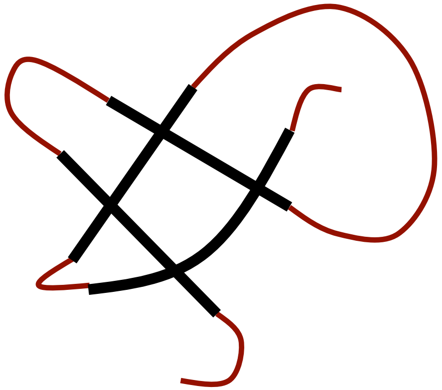
Task Definition: Threading Search

- given:
  - a protein sequence

  - a single template



return: the best alignment of the sequence to the template

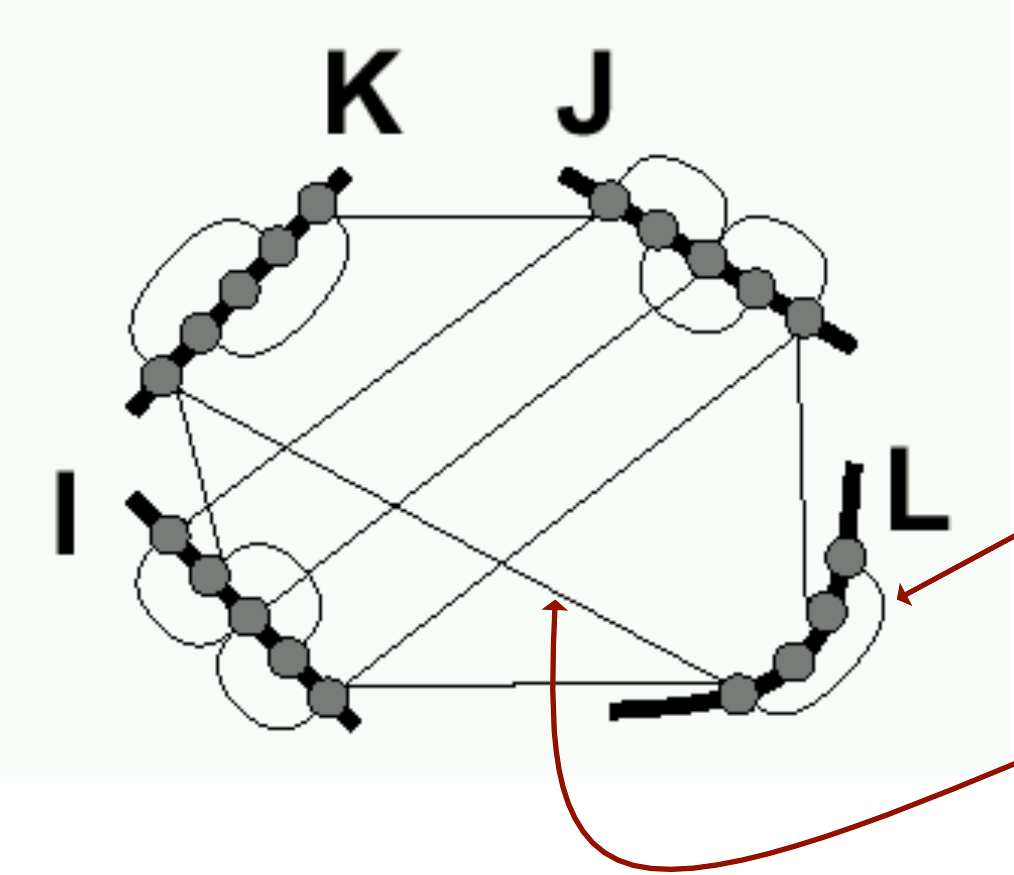


Threading Objective Functions

- possible sequence/template alignments are scored using a specified objective function
- the objective function scores the sequence/structure compatibility between
  - sequence amino acids
  - their corresponding positions in a core template
- it takes into account factors such as
  - a.a. preferences for solvent accessibility
  - a.a. preferences for particular secondary structures
  - interactions among spatially neighboring a.a.’s

Core Template with Interactions

Figure from R. Lathrop et al. Analysis and Algorithms for Protein Sequence-Structure Alignment.



first amino acid in L interacts with last amino acid in L

last amino acid in K interacts with first amino acid in L

- small circles represent amino acid positions
- thin lines indicate interactions represented in model

One Threading

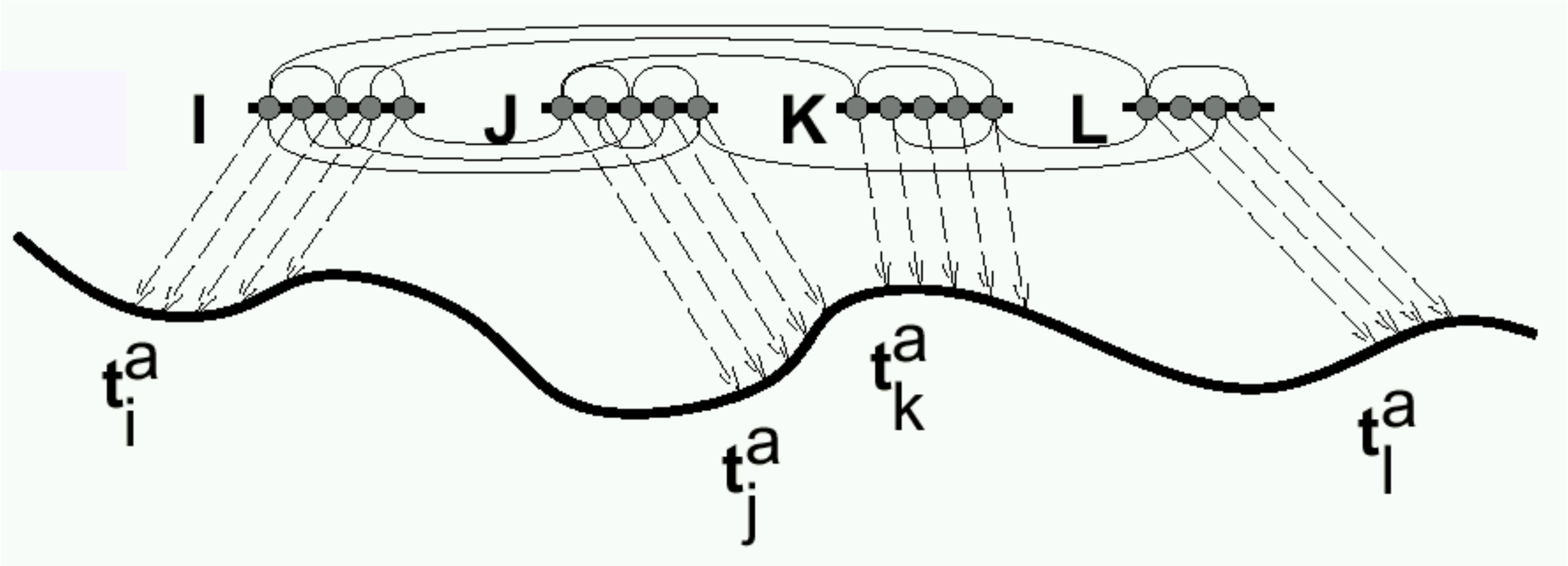
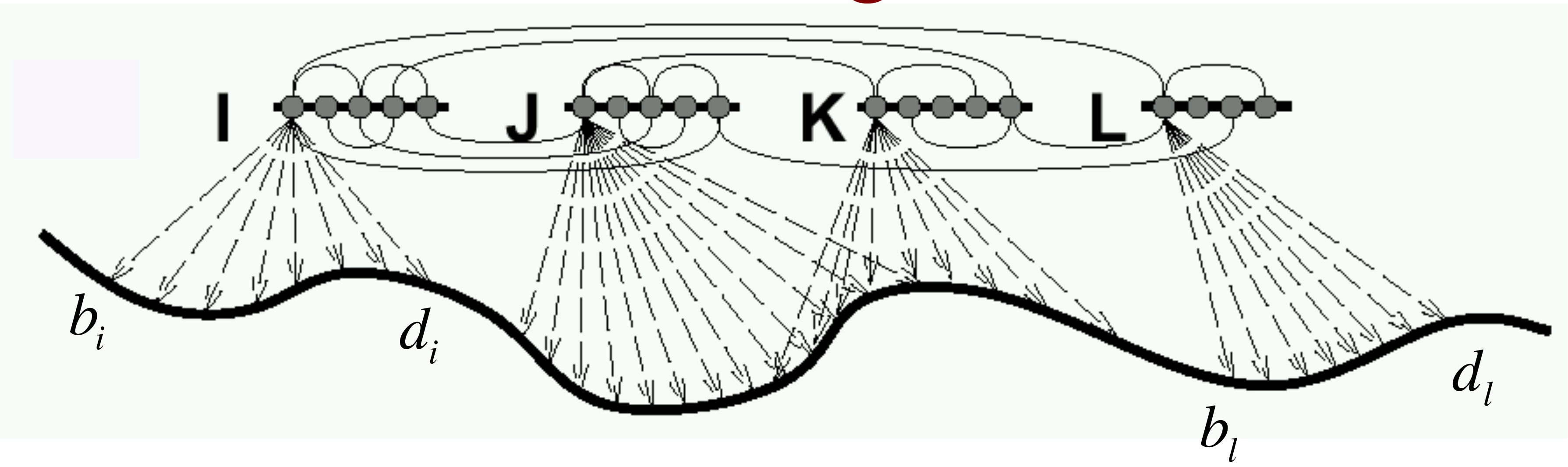


Figure from R. Lathrop et al. Analysis and Algorithms for Protein Sequence-Structure Alignment.

- a threading can be represented as a vector $\vec{t}$, where each element indicates the index of the amino acid placed in the first position of each core segment

Threading Sets



- a set of potential threadings can be represented by bounds on the first position of each core segment

$$T = \left\{ \ \vec{t} \ \mid b_i \le t_i \le d_i, \ b_j \le t_j \le d_j, \ b_k \le t_k \le d_k, \ b_l \le t_l \le d_l \right\}$$

Possible Threadings

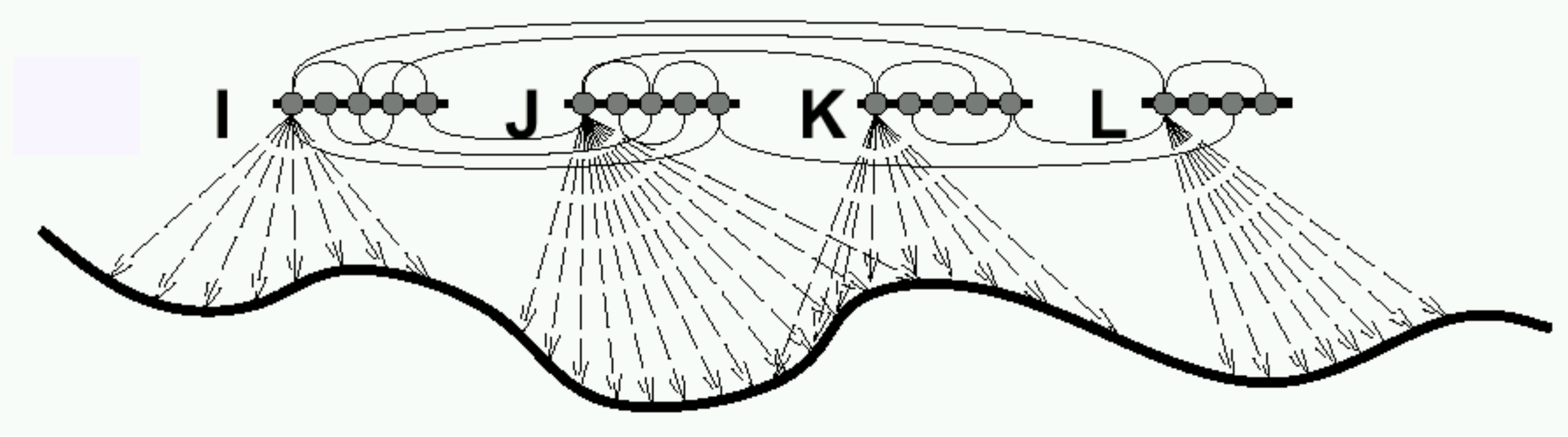


Figure from R. Lathrop et al. Analysis and Algorithms for Protein Sequence-Structure Alignment.

- finding the optimal alignment is NP-hard in the general case where
  - there are variable length gaps between the core segments, and
  - the objective function includes interactions between neighboring (in 3D) amino acids

A General Pairwise Objective Function

- the general objective function with pairwise interactions is:

$$f(\vec{t}\,) = \underbrace{\sum_{i} g_1(i,\, t_i)}_{\text{scores for individual segments}} + \underbrace{\sum_{i} \sum_{j>i} g_2(i,\, j,\, t_i\,,\, t_j)}_{\text{scores for segment interactions}}$$

- We wish to **minimize** this function

Searching the Space of Alignments

- if interaction terms between amino acids are not allowed
  - dynamic programming
    - will find optimal alignment efficiently

- if interaction terms allowed
  - heuristic methods
    - fast
    - might not find the optimal alignment
  - exact methods (e.g. branch & bound)
    - will find the optimal alignment
    - might take exponential time

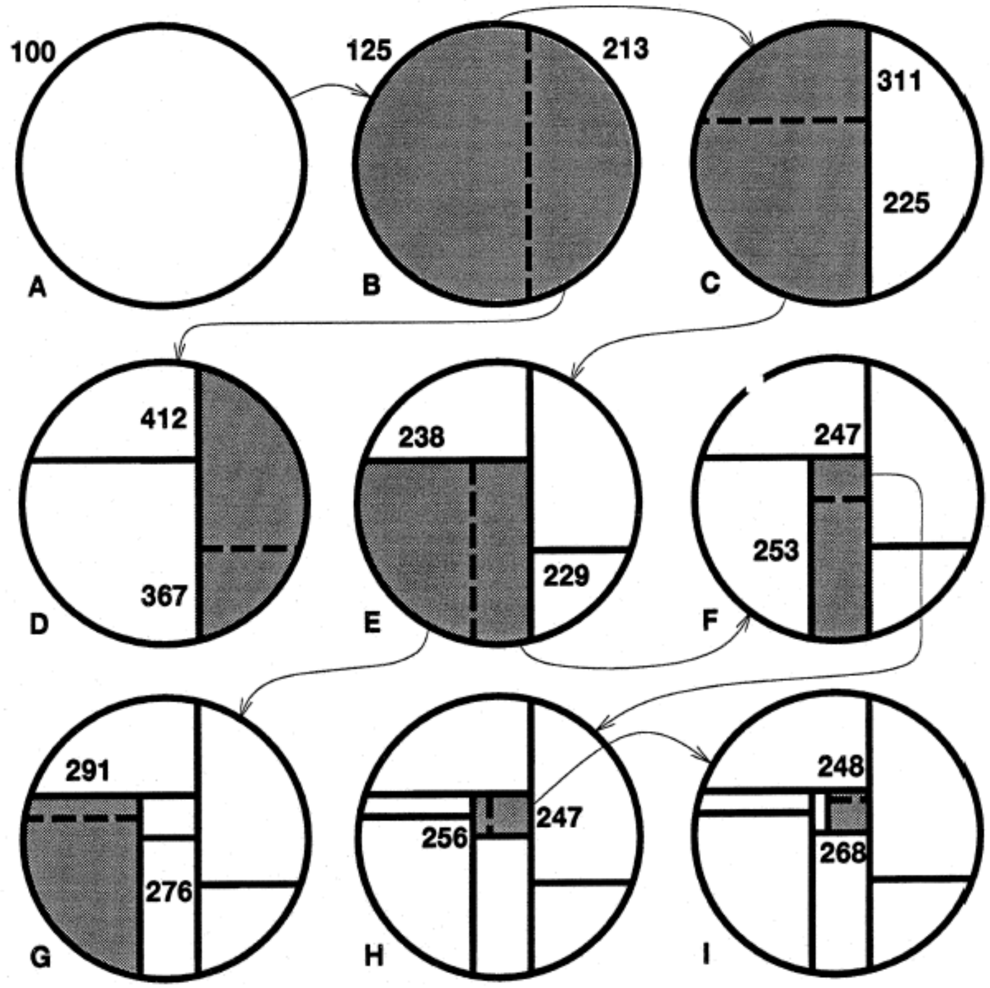
Branch and Bound Abstractly

- 3 components
  - A **data structure** for compactly representing a **set** of potential solutions
    - May be a very large set
  - An **algorithm** for computing a **lower bound** on the score obtained by any member of a set
    - In general, should not explicitly examine all members of the set
  - An **algorithm** for splitting a set into subsets

Branch and Bound Search

```
initialize $Q$ with one entry representing the set of all threadings
repeat
    $l \leftarrow$ set in $Q$ with lowest lower bound
    if $l$ contains only 1 threading
        return $l$
    else
        split $l$ into smaller subsets
        compute lower bound for each subset
        put subsets in $Q$ sorted by lower bound
```

Branch and Bound Illustrated



- a hypothetical branch and bound search
  - each circle illustrates the space of possible threadings
  - solid lines indicate splits made in previous steps
  - dashed lines indicate splits made in current step
  - numbers indicate lower bounds for newly created subsets
  - arrows show the set that has been split

Figure from R. Lathrop and T. Smith. Global Optimum Protein Threading with Gapped Alignment and Empirical Pair Score Functions. *Journal of Molecular Biology* 255:641-665, 1996.

Branch and Bound Search

- there are two key issues to address in instantiating this approach
  - how to compute the lower bound for a set of threadings
  - how to split a threading set into subsets
- these aspects determine the expected efficiency of the branch and bound search

A Simple Lower Bound

$$\left.\min_{\vec{t}\in T} f(\vec{t}\,) = \min_{\vec{t}\in T} \sum_i \left[g_1(i,\, t_i) + \sum_{j>i} g_2(i,\, j,\, t_i\,,\, t_j) \right] \right\} \text{objective function}$$

$$\left.\geq \sum_i \left[\min_{b_i \leq\, x\, \leq\, d_i} g_1(i,x) + \sum_{j>i} \min_{\substack{b_i \leq\, y\, \leq\, d_i \\ b_j \leq\, z\, \leq\, d_j}} g_2(i,j,y,z) \right] \right\} \text{lower bound}$$

- in a nutshell: calculate minimum over each term separately

A Better Lower Bound

$$\min_{\vec{t}\in T} f(\vec{t}) \geq$$

$$\min_{\vec{t}\in T} \sum_{i} \left[g_1(i,\, t_i) + \underbrace{g_2(i-1,\, i,\, t_{i-1},\, t_i)}_{\text{interaction with preceding segment}} + \underbrace{\min_{\vec{u}\in T} \sum_{\substack{j>i+1 \\ j<i-1}} \frac{1}{2} g_2(i,\, j,\, t_i,\, u_j)}_{\text{best case interaction with other segments}} \right]$$

- [Lathrop & Smith, *JMB* ’96]

Splitting a Threading Set

- a threading set is split by choosing
  - a single core segment
  - a split point s_i in the segment

- a simple method
  - split the segment having the widest interval, i.e. $\max_i [d_i - b_i]$
  - choose the split point s_i as the value that results in the lower bound for the set

Branch and Bound: Splitting a Set

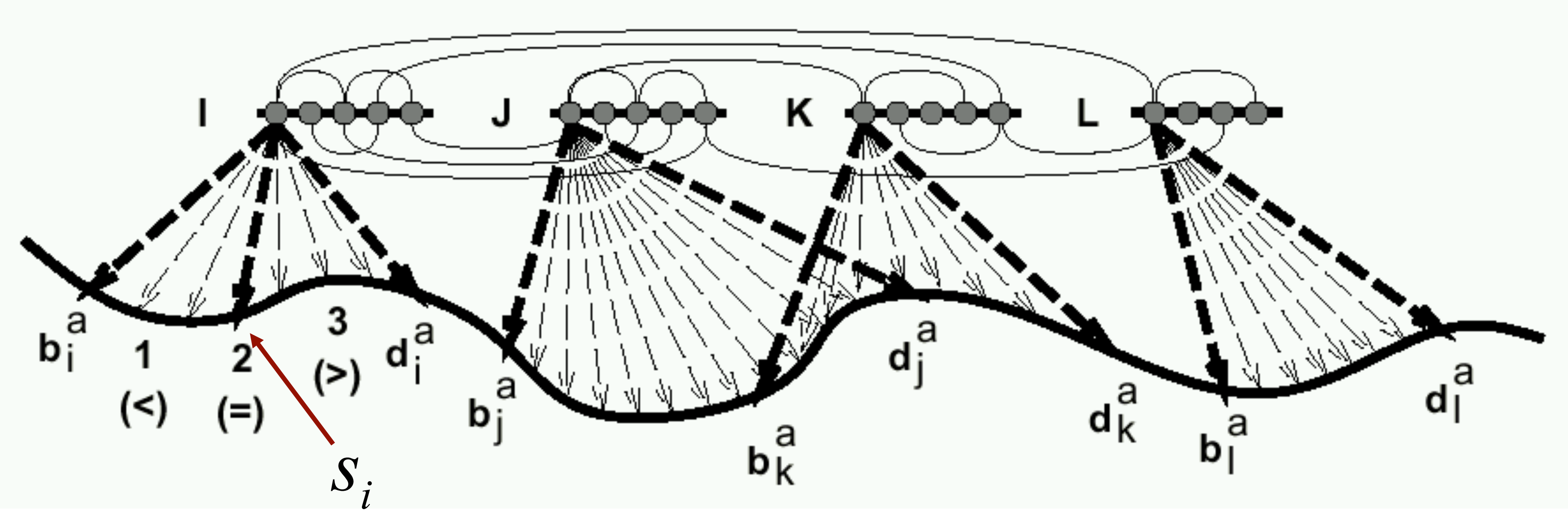


Figure from R. Lathrop et al. Analysis and Algorithms for Protein Sequence-Structure Alignment.

$$T = \left\{ \vec{t} \mid b_i \le t_i \le d_i,\ \ b_j \le t_j \le d_j,\ \ b_k \le t_k \le d_k,\ \ b_l \le t_l \le d_l \right\}$$

splitting into three sets

1 $$T = \left\{ \vec{t} \mid b_i \le t_i < s_i, \quad \cdots \right\}$$

2 $$T = \left\{ \vec{t} \mid t_i = s_i, \quad \cdots \right\}$$

3 $$T = \left\{ \vec{t} \mid s_i < t_i \le d_i, \quad \cdots \right\}$$

Threading Example

Suppose we have three segments (i, j, k), each of which includes three amino acids. For a given sequence there are three possible starting positions for each segment. Suppose that you are given the following values for the scores of the individual segments and the scores for segment interactions.

g1(i,2) = 5
g1(i,3) = 2
g1(i,4) = 8

g1(j,8) = 9
g1(j,9) = 7
g1(j,10) = 6

g1(k,13) = 3
g1(k,14) = 4
g1(k,15) = 1

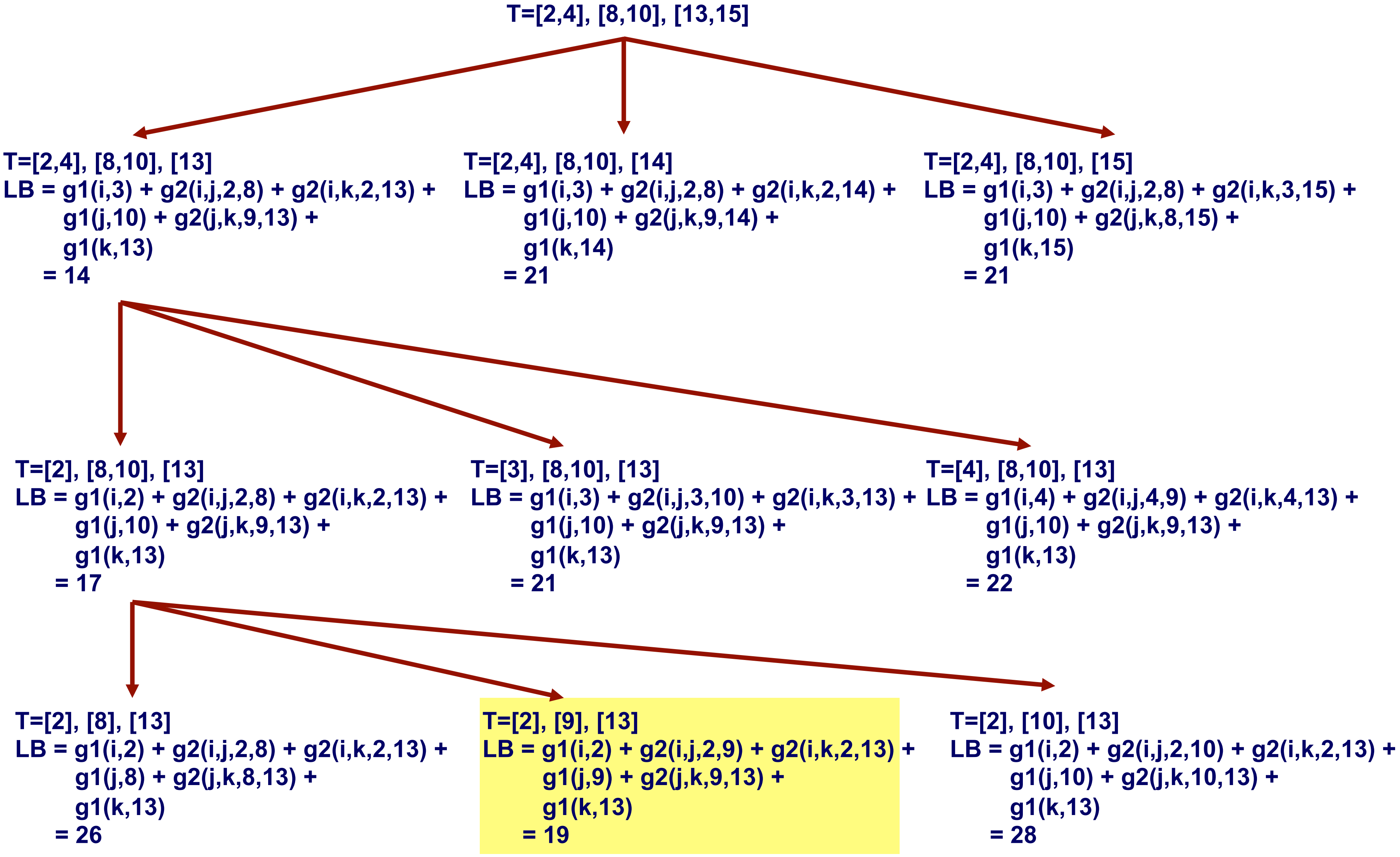
g2(i,j,2,8) = 1
g2(i,j,2,9) = 2
g2(i,j,2,10) = 2
g2(i,j,3,8) = 5
g2(i,j,3,9) = 6
g2(i,j,3,10) = 4
g2(i,j,4,8) = 7
g2(i,j,4,9) = 3
g2(i,j,4,10) = 4

g2(j,k,8,13) = 7
g2(j,k,8,14) = 8
g2(j,k,8,15) = 7
g2(j,k,9,13) = 1
g2(j,k,9,14) = 6
g2(j,k,9,15) = 8
g2(j,k,10,13) = 11
g2(j,k,10,14) = 12
g2(j,k,10,15) = 13

g2(i,k,2,13) = 1
g2(i,k,2,14) = 2
g2(i,k,2,15) = 5
g2(i,k,3,13) = 5
g2(i,k,3,14) = 6
g2(i,k,3,15) = 4
g2(i,k,4,13) = 1
g2(i,k,4,14) = 2
g2(i,k,4,15) = 4

We’ll find the optimal threading using the "simple lower bound" and splitting a set on the segment with the minimal g1 value. When splitting the selected segment, we’ll divide it into three intervals of length one.

Threading Example



T=[2,4], [8,10], [13,15]

T=[2,4], [8,10], [13]
LB = g1(i,3) + g2(i,j,2,8) + g2(i,k,2,13) +
g1(j,10) + g2(j,k,9,13) +
g1(k,13)
= 14

T=[2,4], [8,10], [14]
LB = g1(i,3) + g2(i,j,2,8) + g2(i,k,2,14) +
g1(j,10) + g2(j,k,9,14) +
g1(k,14)
= 21

T=[2,4], [8,10], [15]
LB = g1(i,3) + g2(i,j,2,8) + g2(i,k,3,15) +
g1(j,10) + g2(j,k,8,15) +
g1(k,15)
= 21

T=[2], [8,10], [13]
LB = g1(i,2) + g2(i,j,2,8) + g2(i,k,2,13) +
g1(j,10) + g2(j,k,9,13) +
g1(k,13)
= 17

T=[3], [8,10], [13]
LB = g1(i,3) + g2(i,j,3,10) + g2(i,k,3,13) +
g1(j,10) + g2(j,k,9,13) +
g1(k,13)
= 21

T=[4], [8,10], [13]
LB = g1(i,4) + g2(i,j,4,9) + g2(i,k,4,13) +
g1(j,10) + g2(j,k,9,13) +
g1(k,13)
= 22

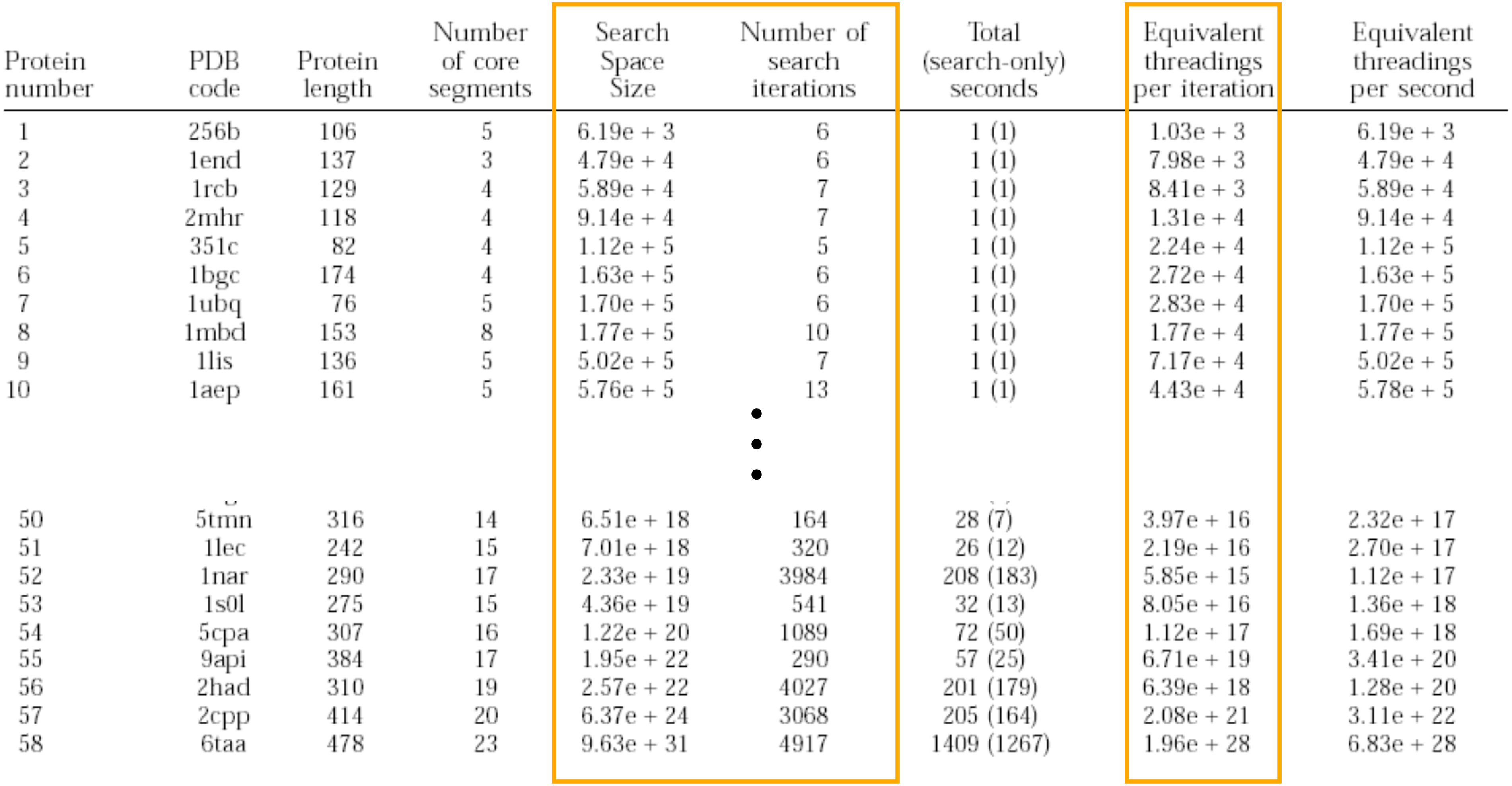
T=[2], [8], [13]
LB = g1(i,2) + g2(i,j,2,8) + g2(i,k,2,13) +
g1(j,8) + g2(j,k,8,13) +
g1(k,13)
= 26

T=[2], [9], [13]
LB = g1(i,2) + g2(i,j,2,9) + g2(i,k,2,13) +
g1(j,9) + g2(j,k,9,13) +
g1(k,13)
= 19

T=[2], [10], [13]
LB = g1(i,2) + g2(i,j,2,10) + g2(i,k,2,13) +
g1(j,10) + g2(j,k,10,13) +
g1(k,13)
= 28

Branch and Bound Efficiency

- 58 proteins threaded against their “native” (i.e. correct) models



Protein number	PDB code	Protein length	Number of core segments	Search Space Size	Number of search iterations	Total (search-only) seconds	Equivalent threadings per iteration	Equivalent threadings per second
1	256b	106	5	6.19e + 3	6	1 (1)	1.03e + 3	6.19e + 3
2	1end	137	3	4.79e + 4	6	1 (1)	7.98e + 3	4.79e + 4
3	1rcb	129	4	5.89e + 4	7	1 (1)	8.41e + 3	5.89e + 4
4	2mhr	118	4	9.14e + 4	7	1 (1)	1.31e + 4	9.14e + 4
5	351c	82	4	1.12e + 5	5	1 (1)	2.24e + 4	1.12e + 5
6	1bgc	174	4	1.63e + 5	6	1 (1)	2.72e + 4	1.63e + 5
7	1ubq	76	5	1.70e + 5	6	1 (1)	2.83e + 4	1.70e + 5
8	1mbd	153	8	1.77e + 5	10	1 (1)	1.77e + 4	1.77e + 5
9	1lis	136	5	5.02e + 5	7	1 (1)	7.17e + 4	5.02e + 5
10	1aep	161	5	5.76e + 5	13	1 (1)	4.43e + 4	5.78e + 5
					⋮			
50	5tmn	316	14	6.51e + 18	164	28 (7)	3.97e + 16	2.32e + 17
51	1lec	242	15	7.01e + 18	320	26 (12)	2.19e + 16	2.70e + 17
52	1nar	290	17	2.33e + 19	3984	208 (183)	5.85e + 15	1.12e + 17
53	1s01	275	15	4.36e + 19	541	32 (13)	8.05e + 16	1.36e + 18
54	5cpa	307	16	1.22e + 20	1089	72 (50)	1.12e + 17	1.69e + 18
55	9api	384	17	1.95e + 22	290	57 (25)	6.71e + 19	3.41e + 20
56	2had	310	19	2.57e + 22	4027	201 (179)	6.39e + 18	1.28e + 20
57	2cpp	414	20	6.37e + 24	3068	205 (164)	2.08e + 21	3.11e + 22
58	6taa	478	23	9.63e + 31	4917	1409 (1267)	1.96e + 28	6.83e + 28

Table from R. Lathrop and T. Smith, *Journal of Molecular Biology* 255:641-665, 1996.