Adaptive Parsing

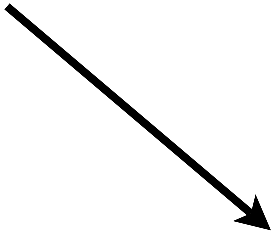
Adrian Sampson
Harvey Mudd College

December 18, 2007
CS152, Fall 2007; Professor Robert Keller

Parsing

Flat text

```
<hwlist>
<name>Problem Set #1</name><due>09/11/2007</due>
<name>Problem Set #2</name><due>09/18/2007</due>
<name>Problem Set #3</name><due>09/25/2007</due>
<name>Problem Set #4</name><due>10/02/2007</due>
<name>Problem Set #5</name><due>10/18/2007</due>
<name>Problem Set #6</name><due>10/30/2007</due>
<name>Problem Set #7</name><due>11/06/2007</due>
<name>Problem Set #8</name><due>11/20/2007</due>
<name>Problem Set #9</name><due>11/27/2007</due>
<name>Problem Set #10</name><due>12/04/2007</due>
</hwlist>
```



Data structure

```
[['1', '09/11'], ['2', '09/18'], ['3',
'09/25'], ['4', '10/02'], ['5', '10/18'],
['6', '10/30'], ['7', '11/06'], ['8',
'11/20'], ['9', '11/27'], ['10', '12/04']]
```

Adaptive Parsing

Flat text

```
<hwlist>
<name>Problem Set #1</name><due>09/11/2007</due>
<name>Problem Set #2</name><due>09/18/2007</due>
<name>Problem Set #3</name><due>09/25/2007</due>
<name>Problem Set #4</name><due>10/02/2007</due>
<name>Problem Set #5</name><due>10/18/2007</due>
<name>Problem Set #6</name><due>10/30/2007</due>
<name>Problem Set #7</name><due>11/06/2007</due>
<name>Problem Set #8</name><due>11/20/2007</due>
<name>Problem Set #9</name><due>11/27/2007</due>
<name>Problem Set #10</name><due>12/04/2007</due>
</hwlist>
```

(minimal syntax information)

Data structure

```
[['1', '09/11'], ['2', '09/18'], ['3',
'09/25'], ['4', '10/02'], ['5', '10/18'],
['6', '10/30'], ['7', '11/06'], ['8',
'11/20'], ['9', '11/27'], ['10', '12/04']]
```

Neural Problem

Find patterns in text representing delimiters

$\Rightarrow$ Cluster substrings

$\Rightarrow$ Vector quantization (strings are vectors)

$\Rightarrow$ Competitive learning algorithm

Substrings Are Input Vectors

```
<hwlist>
<name>Problem Set #1</name><due>09/11/2007</due>
<name>Problem Set #2</name><due>09/18/2007</due>
<name>Problem Set #3</name><due>09/25/2007</due>
<name>Problem Set #4</name><due>10/02/2007</due>
<name>Problem Set #5</name><due>10/18/2007</due>
<name>Problem Set #6</name><due>10/30/2007</due>
<name>Problem Set #7</name><due>11/06/2007</due>
<name>Problem Set #8</name><due>11/20/2007</due>
<name>Problem Set #9</name><due>11/27/2007</due>
<name>Problem Set #10</name><due>12/04/2007</due>
</hwlist>
```

$$\xrightarrow{\quad w \quad}$$

$$\overbrace{\qquad\qquad}^{w}$$

```
<hwlis
hwlist
wlist>
list>¬
ist>¬<
st>¬<n
t>¬<na
>¬<nam
¬<name
<name>
name>P
ame>Pr
me>Pro
e>Prob
>Probl
Proble
```

$$\vdots$$

Euclidean Distance of Byte Strings

aaa
$(97, 97, 97)$

aba
$(97, 98, 97)$

aza
$(97, 122, 97)$

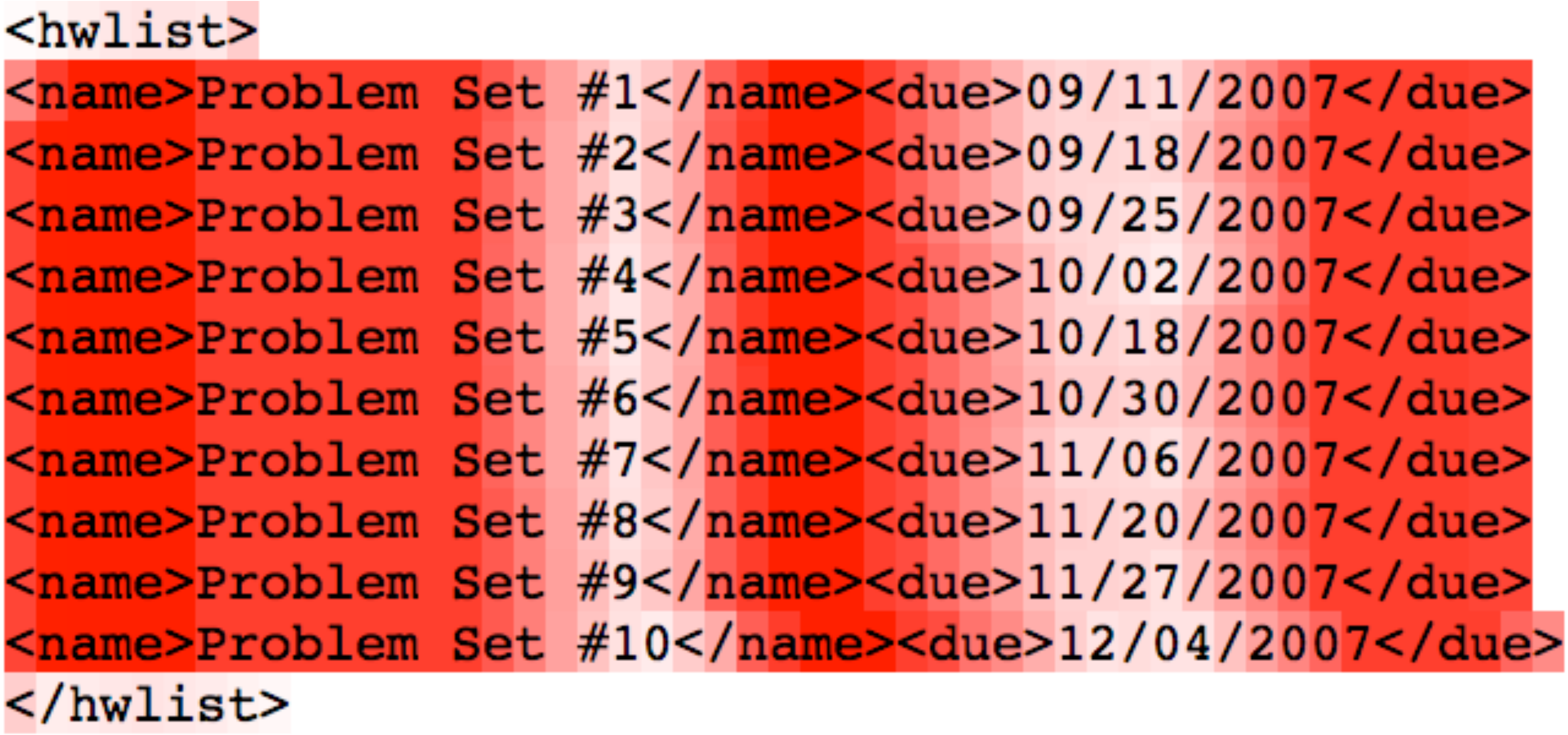
Euclidean Distance of One-Hot Strings

aaa

aza

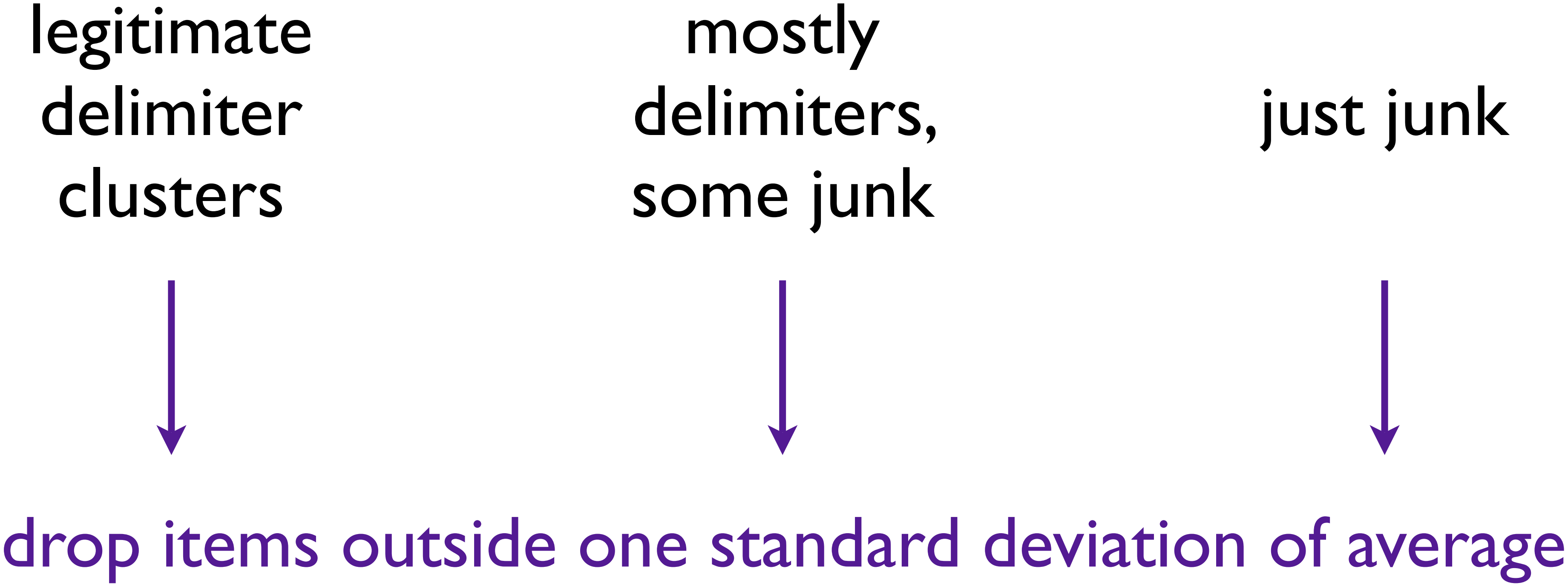
aba

$$(0, 0, 0, 0, 0, 0, 0, \ldots, 0, 1, 0, \ldots, 0, 0, 0)$$



```
<hwlist>
<name>Problem Set #1</name><due>09/11/2007</due>
<name>Problem Set #2</name><due>09/18/2007</due>
<name>Problem Set #3</name><due>09/25/2007</due>
<name>Problem Set #4</name><due>10/02/2007</due>
<name>Problem Set #5</name><due>10/18/2007</due>
<name>Problem Set #6</name><due>10/30/2007</due>
<name>Problem Set #7</name><due>11/06/2007</due>
<name>Problem Set #8</name><due>11/20/2007</due>
<name>Problem Set #9</name><due>11/27/2007</due>
<name>Problem Set #10</name><due>12/04/2007</due>
</hwlist>
```

Filtering Output



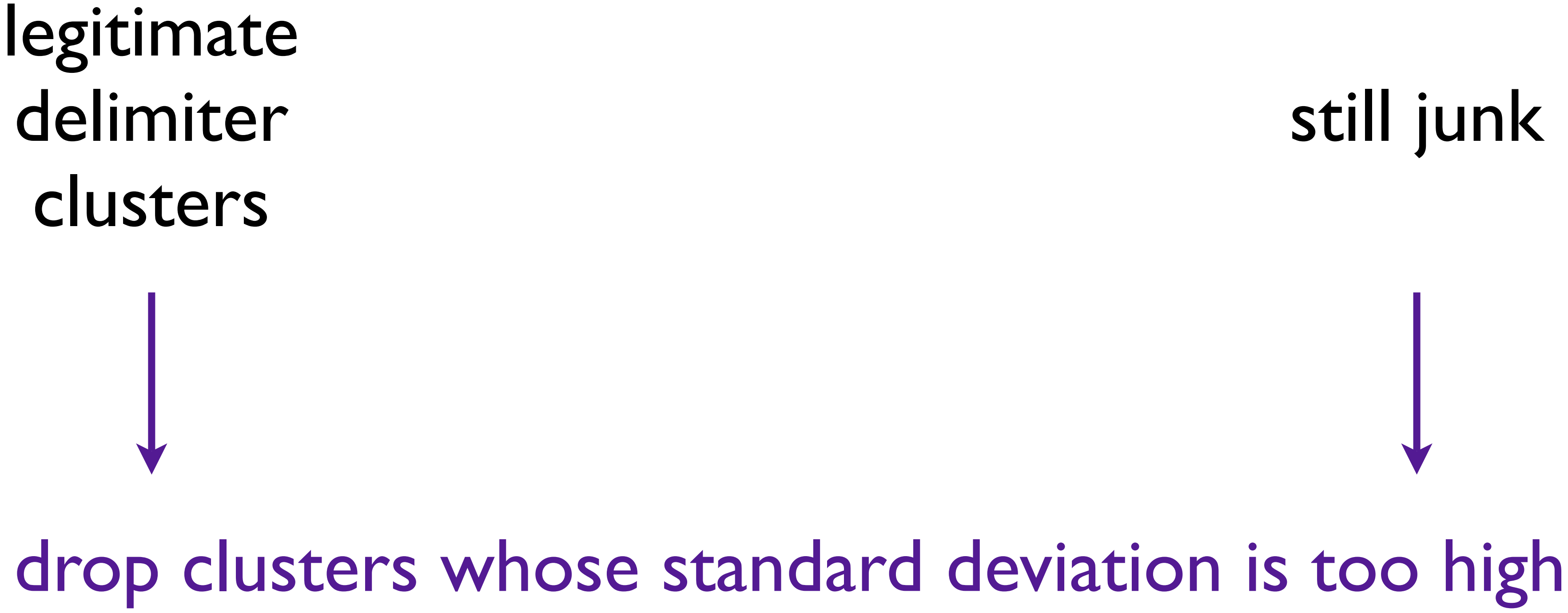
legitimate delimiter clusters

mostly delimiters, some junk

just junk

drop items outside one standard deviation of average

Filtering Output



legitimate delimiter clusters

still junk

drop clusters whose standard deviation is too high

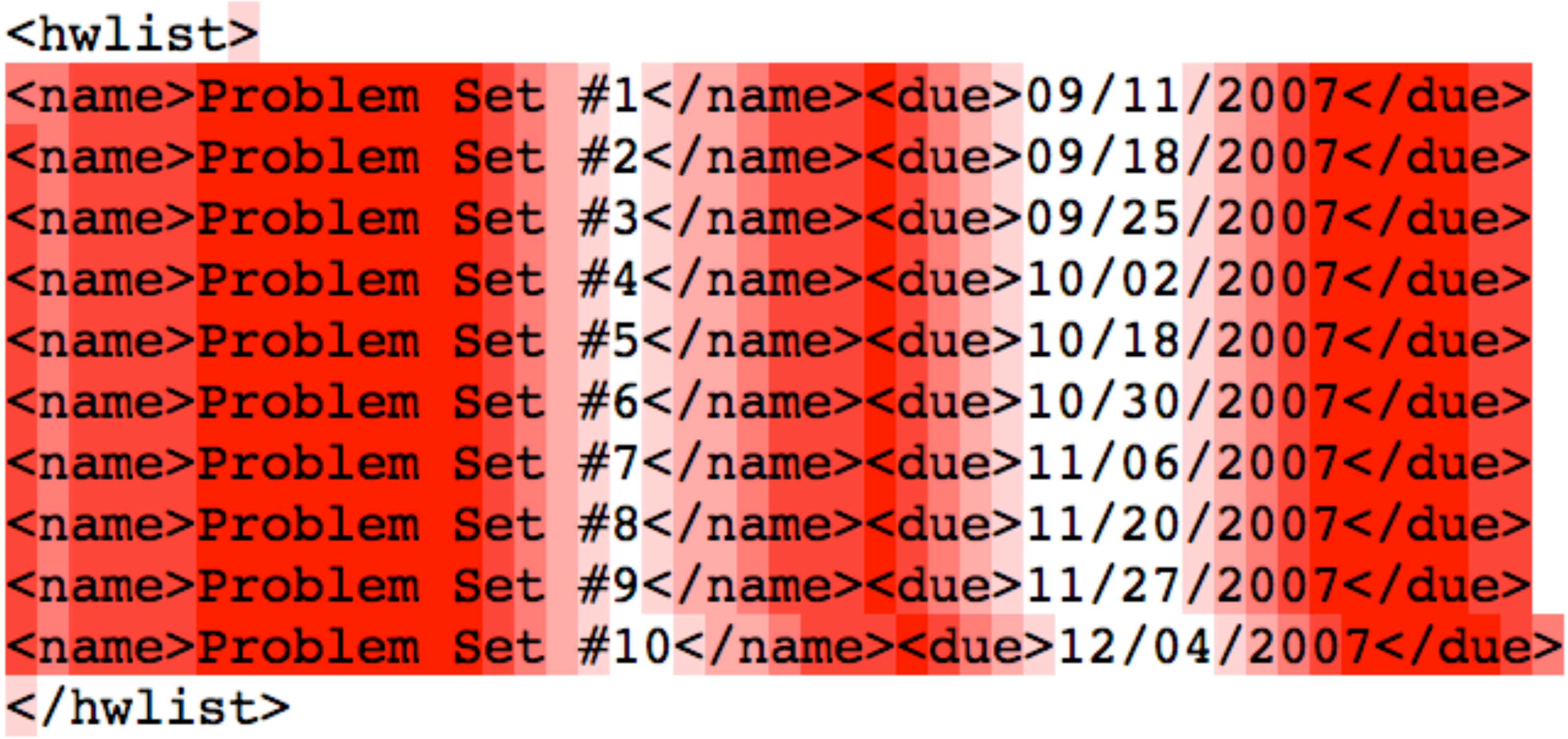
Filtering Output

legitimate
delimiter
clusters

Shifting Output

Delimiters will repeat n times where n is the number of vectors

$\Rightarrow$ Prefer vectors whose popularity is *uniform*



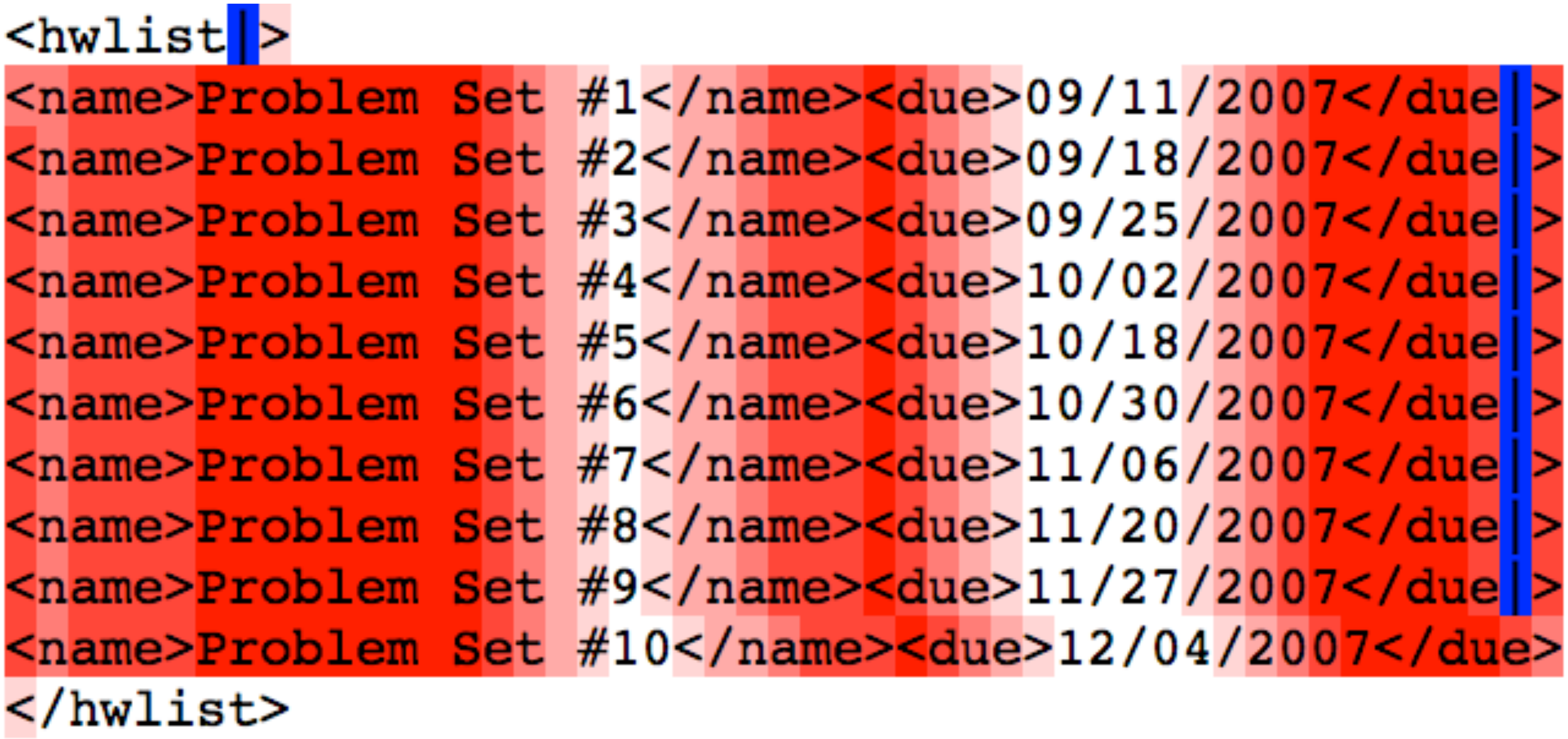
```
<hwlist>
<name>Problem Set #1</name><due>09/11/2007</due>
<name>Problem Set #2</name><due>09/18/2007</due>
<name>Problem Set #3</name><due>09/25/2007</due>
<name>Problem Set #4</name><due>10/02/2007</due>
<name>Problem Set #5</name><due>10/18/2007</due>
<name>Problem Set #6</name><due>10/30/2007</due>
<name>Problem Set #7</name><due>11/06/2007</due>
<name>Problem Set #8</name><due>11/20/2007</due>
<name>Problem Set #9</name><due>11/27/2007</due>
<name>Problem Set #10</name><due>12/04/2007</due>
</hwlist>
```

Delimiting Records

- Find first contiguous sequence of delimiters
  - Collect sequence of cluster IDs

```
<name,  name>,  ame>P,  me>Pr,  ...
```

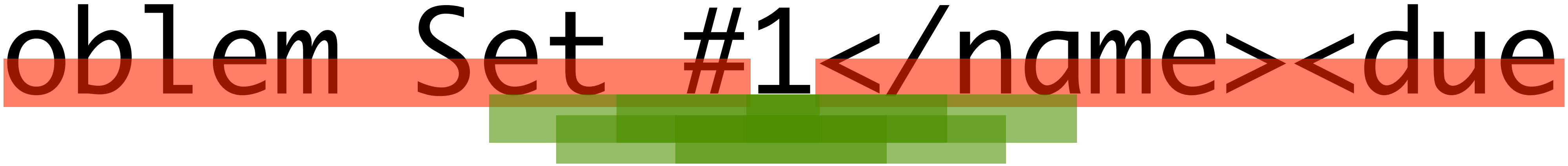
- Search for delimiter sequences whose IDs match this sequence



```
<hwlist>
<name>Problem Set #1</name><due>09/11/2007</due>
<name>Problem Set #2</name><due>09/18/2007</due>
<name>Problem Set #3</name><due>09/25/2007</due>
<name>Problem Set #4</name><due>10/02/2007</due>
<name>Problem Set #5</name><due>10/18/2007</due>
<name>Problem Set #6</name><due>10/30/2007</due>
<name>Problem Set #7</name><due>11/06/2007</due>
<name>Problem Set #8</name><due>11/20/2007</due>
<name>Problem Set #9</name><due>11/27/2007</due>
<name>Problem Set #10</name><due>12/04/2007</due>
</hwlist>
```

Finding Fields

$$w = 5$$



```
oblem Set #1</name><due
```

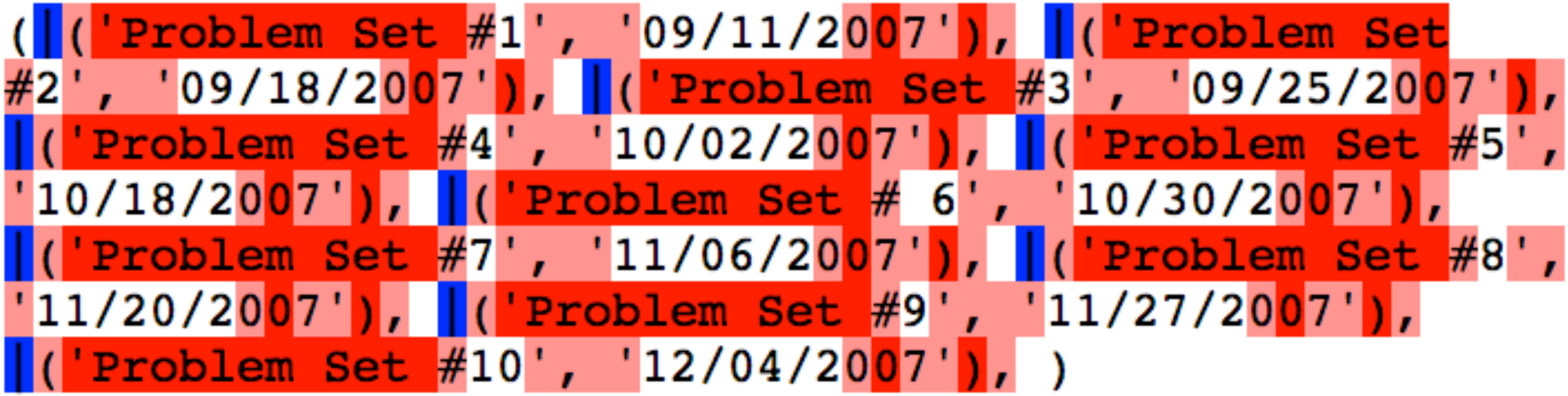
w vectors on left and w vectors on right of each field are unpopular

$\Rightarrow$ fields marked by w contiguous unpopular vectors

```
[['1', '09/11'], ['2', '09/18'],
['3', '09/25'], ['4', '10/02'],
['5', '10/18'], ['6', '10/30'],
['7', '11/06'], ['8', '11/20'],
['9', '11/27'], ['10', '12/04']]
```

Awesome!

Another Example



```
( ('Problem Set #1', '09/11/2007'),  ('Problem Set
#2', '09/18/2007'),  ('Problem Set #3', '09/25/2007'),
('Problem Set #4', '10/02/2007'),  ('Problem Set #5',
'10/18/2007'),  ('Problem Set # 6', '10/30/2007'),
('Problem Set #7', '11/06/2007'),  ('Problem Set #8',
'11/20/2007'),  ('Problem Set #9', '11/27/2007'),
('Problem Set #10', '12/04/2007'), )
```

```
[['1', '09/11/2', ' '],
 ['2', '09/18/2', ' '],
 ['3', '09/25/2', ' '],
 ['4', '10/02/2', ' '],
 ['5', '10/18/2', ' '],
 [' 6', '10/30/2', ' '],
 ['7', '11/06/2', ' '],
 ['8', '11/20/2', ' '],
 ['9', '11/27/2', ' '],
 ['10', '12/04/2']]
```

Observations

- Choosing w is the hardest (least automatable) task
- Delicate balance: tolerance to noise vs. sensitivity to fields

Future Work

- Implement competitive learning myself
    - Use distance metric that incorporates insertions & deletions

… faster than one-hot strings?

Future Work

- Use sequence clustering work from bioinformatics

Future Work

- Runtime
    - Keep “training” to parse future modifications/additions to document easily