Application-Aware Data Collection in Wireless Sensor Networks

Fang Xiaolin, Gao Hong, Li Jianzhong
Harbin Institute of Technology
Li Yingshu
Georgia State University
Outline

- Existing work
- Our problem
- An approximation algorithm
- A special instance
- Simulations
Existing work

- Multi-application based data collection
  - Data sharing [9]
  - Sample as less data as possible

Existing work studies data point sampling
Our problem

- Multi-application based data collection
- Sample data for a continuous interval
  - Acoustic, video information [10], [11]
  - Vibration measurement [12], [13], [14]
  - Speed information [15]

Sample an interval
Problem definition

Given a set of $n$ tasks $T$, each task $T_i$ is denoted as $T_i = \langle b_i, e_i, l_i \rangle$

- Find a continuous sub-interval $I_i$ for each task $T_i$ so that

\[
\min \left| \bigcup_{i=1}^{n} I_i \right|
\]

Define $I \cup I'$ as the union $[1, 5] \cup [3, 7] = [1, 7]$
Problem complexity

- Non-linear non-convex optimization problem
- Nonlinear integer programming problem, if $b_i, e_i, l_i$ are regarded as integers
Greedy algorithm overview

1. Sort by end times
2. Find task set P overlap with first
3. Find solution for P, and Remove
4. Back to step 2
Greedy algorithm overview

1. Sort by end times
2. Find task set P overlap with first
3. Find solution for P, and Remove
4. Back to step 2

T1
T2
T3
T4
T5
Greedy algorithm overview

1. Sort by end times
2. Find task set P overlap with first
3. Find solution for P, and Remove
4. Back to step 2

\[ T_1, T_2, T_3, T_4, T_5 \]
Greedy algorithm overview

1. Sort by end times
2. Find task set P overlap with first
3. Find solution for P, and Remove
4. Back to step 2
Find solution for P

- Compute \([s,e]\) for the tasks overlap with each other

\[
s = \min_{i=1}^{n}\{e_i - l_i\}
\]

\[
e = \max\{\max_{i=1}^{n}\{b_i + l_i\}, \max_{i=1}^{n}\{s + l_i\}, \min_{i=1}^{n}\{e_i\}\}
\]

\([s,e]=[5,14]\)
Approximation algorithm analysis

- Approximation ratio is 2
- Time complexity is $O(n^2)$
A special instance

- General problem $T_i = \langle b_i, e_i, l_i \rangle$
- Special instance $T_i = \langle b_i, e_i, l \rangle$
- The data lengths are the same

Can be solved in $O(n^2)$
Algorithm overview

1. Sort by the end times
2. Remove tasks cover other tasks
3. Dynamic programming

Does not affect the result
Algorithm overview

1. Sort by the end times
2. Remove tasks cover other tasks
3. Dynamic programming

\[ I(i, j) = \begin{cases} [e_i' - l, e_i'] & \text{if } b_j' \leq e_i' - l, \\ [e_i' - l, b_j' + l] & \text{if } e_i' - l < b_j' < e_i' \end{cases} \]

\[ g(i) = \begin{cases} \arg \min_{i \leq x \leq m} \{|I(i, x) \uplus f(x + 1)|\} & 1 \leq i < m \\ m & i = m \end{cases} \]

\[ f(i) = \begin{cases} I(i, g(i)) \uplus f(g(i) + 1) & 1 \leq i < m \\ [e_m' - l, e_m'] & i = m \\ \emptyset & i > m \end{cases} \]
Algorithm overview

1. Sort by the end times
2. Remove tasks cover other tasks
3. Dynamic programming

Computing $x(i,j)$, then $x(1,n)$ is the result

<table>
<thead>
<tr>
<th>$x(i,j)$</th>
<th>[s,e]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x(1,1)$</td>
<td>[3,7]</td>
</tr>
<tr>
<td>$x(2,2)$</td>
<td>[5,9]</td>
</tr>
<tr>
<td>$x(3,3)$</td>
<td>[12,16]</td>
</tr>
<tr>
<td>$x(1,2)$</td>
<td>[3,8]</td>
</tr>
<tr>
<td>$x(2,3)$</td>
<td>[5,10]</td>
</tr>
</tbody>
</table>
Algorithm overview

1. Sort by the end times
2. Remove tasks cover other tasks
3. Dynamic programming
   Computing $x(i,j)$, then $x(1,n)$ is the result

<table>
<thead>
<tr>
<th>$x(i,j)$</th>
<th>[s,e]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x(1,1)$</td>
<td>[3,7]</td>
</tr>
<tr>
<td>$x(2,2)$</td>
<td>[5,9]</td>
</tr>
<tr>
<td>$x(3,3)$</td>
<td>[12,16]</td>
</tr>
<tr>
<td>$x(1,2)$</td>
<td>[3,8]</td>
</tr>
<tr>
<td>$x(2,3)$</td>
<td>[5,10]</td>
</tr>
</tbody>
</table>
Algorithm overview

- 1. Sort by the end times
- 2. Remove tasks cover other tasks
- 3. Dynamic programming

Computing $x(i,j)$, then $x(1,n)$ is the result

<table>
<thead>
<tr>
<th>$x(i,j)$</th>
<th>$[s,e]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x(1,1)$</td>
<td>[3,7]</td>
</tr>
<tr>
<td>$x(2,2)$</td>
<td>[5,9]</td>
</tr>
<tr>
<td>$x(3,3)$</td>
<td>[12,16]</td>
</tr>
<tr>
<td>$x(1,2)$</td>
<td>[3,8]</td>
</tr>
<tr>
<td>$x(2,3)$</td>
<td>[5,10]</td>
</tr>
</tbody>
</table>
**Algorithm overview**

1. Sort by the end times
2. Remove tasks cover other tasks
3. Dynamic programming

Computing $x(i,j)$, then $x(1,n)$ is the result

<table>
<thead>
<tr>
<th>$x(i,j)$</th>
<th>[s,e]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x(1,1)$</td>
<td>[3,7]</td>
</tr>
<tr>
<td>$x(2,2)$</td>
<td>[5,9]</td>
</tr>
<tr>
<td>$x(3,3)$</td>
<td>[12,16]</td>
</tr>
<tr>
<td>$x(1,2)$</td>
<td>[3,8]</td>
</tr>
<tr>
<td>$x(2,3)$</td>
<td>[5,10]</td>
</tr>
</tbody>
</table>

$x(1,3) = \min \left\{ x(1,1) \cup x(2,3), x(1,2) \cup x(3,3) \right\}$
Simulation

- Tossim
- Four cases
- Tasks are from multi-applications
- Each application consists of periodical tasks
Simulation

- short sampling interval lengths
Simulation

- longer sampling interval lengths
Simulation

- Different Window size

![Bar chart showing data amount per unit time for different window sizes, comparing optimal, greedy, and naive methods.](image)
Simulation

- Data loss

![Bar chart showing data loss comparison between naive and greedy methods for different node counts.](chart.png)
THANK YOU! ^_^