Location Prediction for Indoor Sensor Networks

Dong Min Kim

Yonsei Univeristy

August 23, 2008

Contents

I Background

II Motivation

III System model

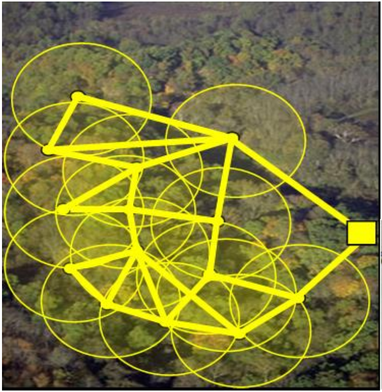
IV Proposed scheme

V Experiment

VI Conclusion

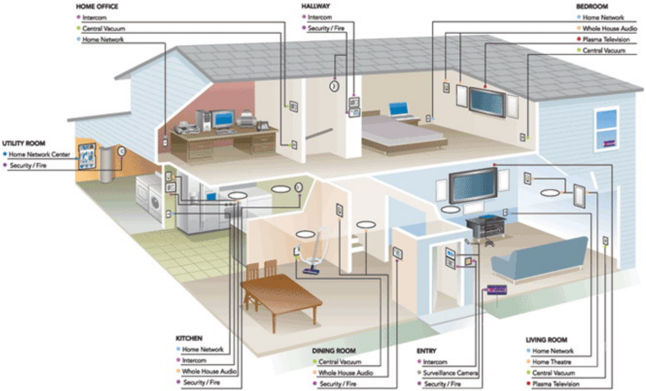
Wireless Sensor Network

- data sensing, processing, wireless communication networks
- small size, energy efficient, low cost, robust radio



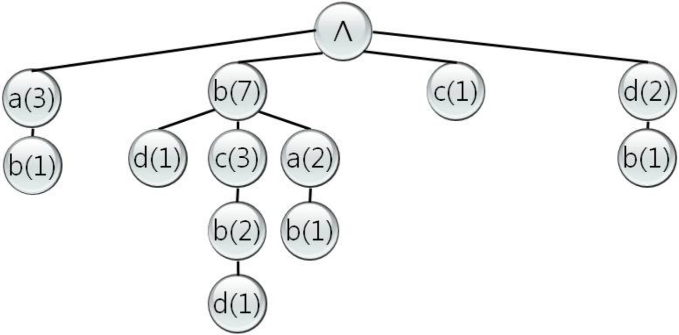
Intelligent Building Application

- ubiquitous computing environments, smart home, healthcare, emergency
- security, automation,



Location Prediction

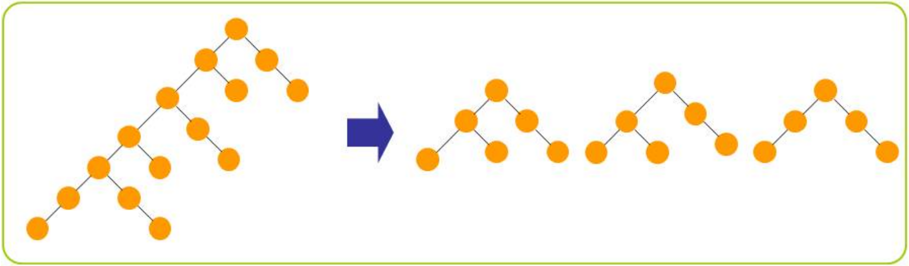
- LeZi-Update[1]
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a
  - a / b / c / bd / bc / ba / bcb / ab / d / bcbd / bab / db / a



[1] A. Roy, S. Das, K. Basu, "A Predictive Framework for Location-Aware Resource Management in Smart Homes," IEEE Transactions on Mobile Computing, vol. 6, no. 11, November 2007.

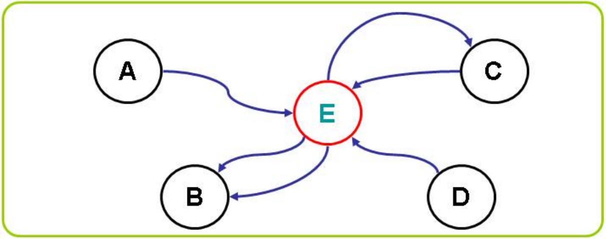
Motivation

- Exploit tree structure (very efficient to represent paths)
- For low complexity, sliding fixed-width window
- Indoor environment moving pattern is composed of several short segments



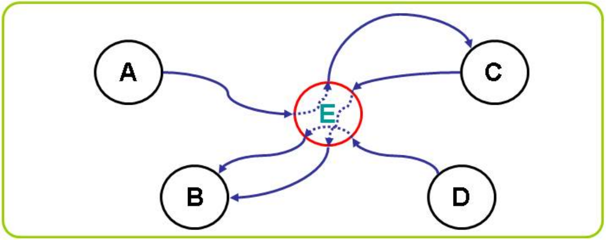
System model

- Binary motion sensor
- Individual tracking may not be applicable
  - Moving inhabitants are not equipped with identifying device
  - Ex) RFID Tag, PDA, etc.
- Group tracking is possible
  - We know the number of inhabitants currently in home
  - Only know about the multiple position



System model

- Binary motion sensor
- Individual tracking may not be applicable
  - Moving inhabitants are not equipped with identifying device
  - Ex) RFID Tag, PDA, etc.
- Group tracking is possible
  - We know the number of inhabitants currently in home
  - Only know about the multiple position

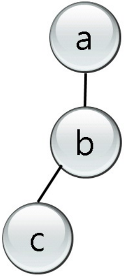


k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a

k-level Tree based Prediction

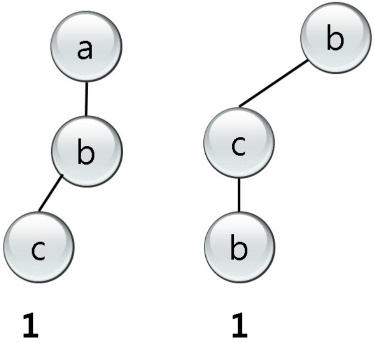
- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a



1

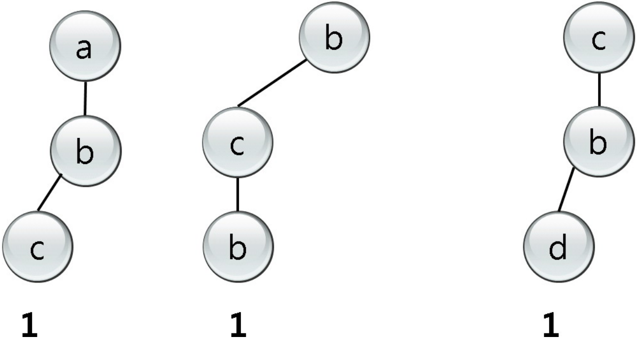
k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a



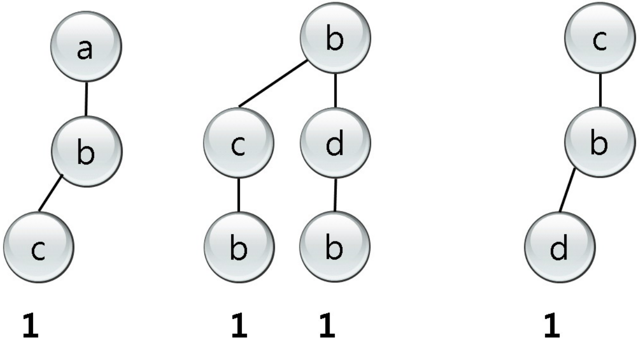
k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a



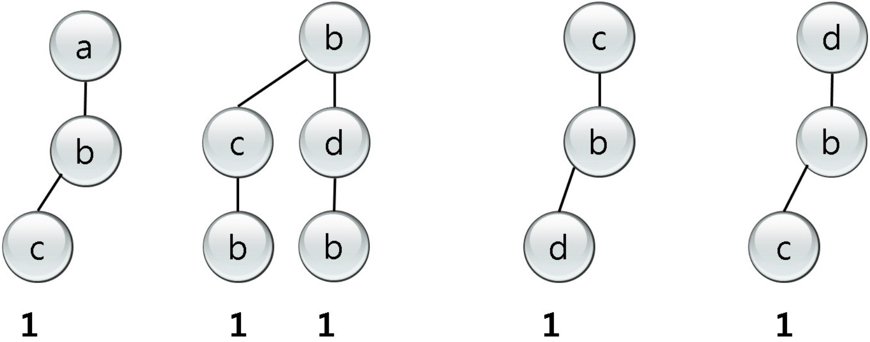
k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a



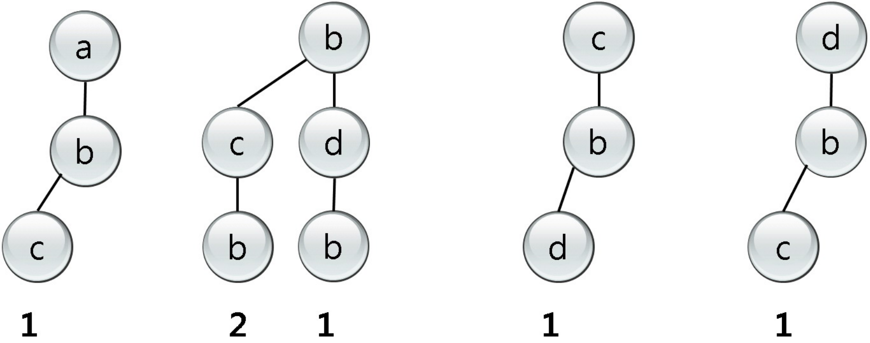
k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a



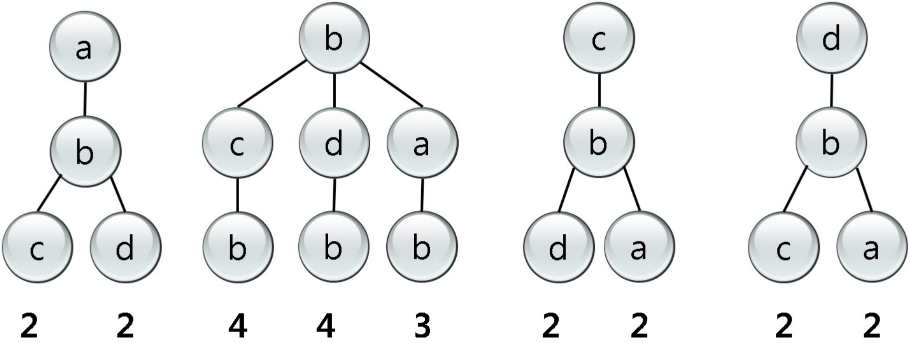
k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a



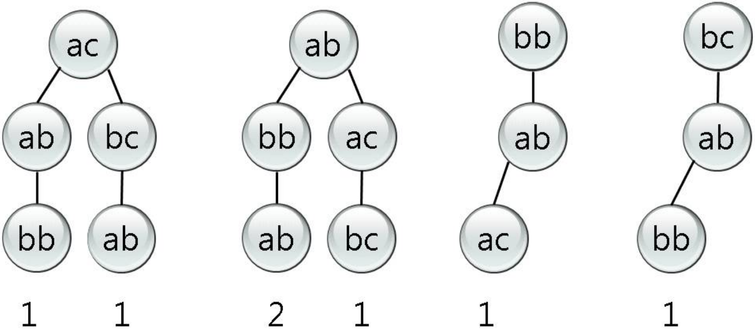
k-level Tree based Prediction

- $K = 3$
- Sequence: a b c b d b c b a b c b a b d b c b d b a b d b a

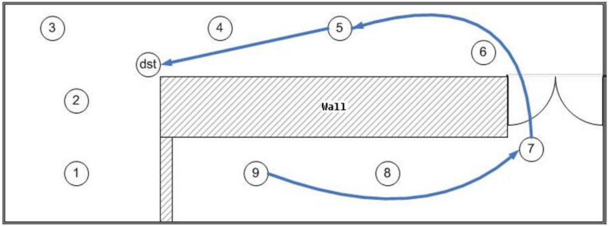


Multiple Inhabitants Case

- 2 inhabitant, $K = 3$ case
  - Conceptually Sensory data is 2-tuple sequences
  - Sequence: (a,c) (a,b) (b,b) (a,b) (a,c) (b,c) (a,b) (b,b) (a,b)



Motion-Detection Sensor Testbed

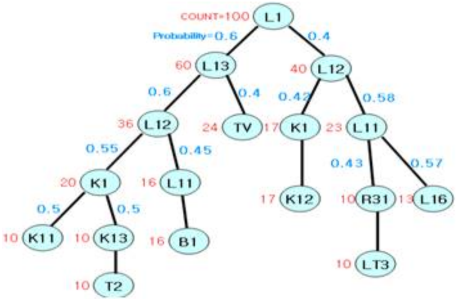


Location Prediction Algorithm Implementation

- right path

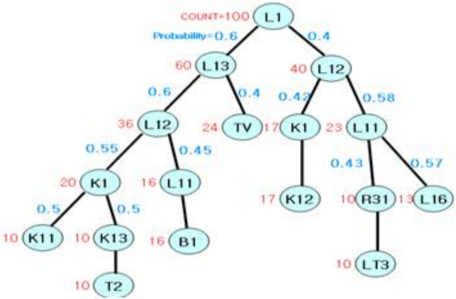
Location Prediction Algorithm Implementation

- right path
    - make data structure



Location Prediction Algorithm Implementation

- right path
  - make data structure



- complicate, time consuming...

Location Prediction Algorithm Implementation

- shortcut

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...

Location Prediction Algorithm Implementation

- shortcut
    - exploit characteristics
- Prediction server receives sequence of sensor ID
    - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants
  - 1 inhabitant: A B C H I F C F I E F I C F I E F I G H I C...

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants
  - 1 inhabitant: A B C H I F C F I E F I C F I E F I G H I C...
  - 2 inhabitatns: AB CH IF CF IE FI CF IE FI GH IC FI HI FC...

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants
  - 1 inhabitant: A B C H I F C F I E F I C F I E F I G H I C...
  - 2 inhabitatns: AB CH IF CF IE FI CF IE FI GH IC FI HI FC...
- All about text processing!

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants
  - 1 inhabitant: A B C H I F C F I E F I C F I E F I G H I C...
  - 2 inhabitatns: AB CH IF CF IE FI CF IE FI GH IC FI HI FC...
- All about text processing!
- use Perl;

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants
  - 1 inhabitant: A B C H I F C F I E F I C F I E F I G H I C...
  - 2 inhabitatns: AB CH IF CF IE FI CF IE FI GH IC FI HI FC...
- All about text processing!
- use Perl;
- Searching CPAN!

Location Prediction Algorithm Implementation

- shortcut
  - exploit characteristics
- Prediction server receives sequence of sensor ID
  - ABCHIFCFIEFICFIEFIGHICFIHIFCFEGHIEFIGHIHIFEF...
- Grouping sequence by number of inhabitants
  - 1 inhabitant: A B C H I F C F I E F I C F I E F I G H I C...
  - 2 inhabitatns: AB CH IF CF IE FI CF IE FI GH IC FI HI FC...
- All about text processing!
- use Perl;
- Searching CPAN!
  - http://search.cpan.org/∼vlado/Text-Ngrams-1.9/Ngrams.pm

use Text::Ngrams;

- sequence.txt: A B C H I F C F I E F I C F I

```
$ perl ngram.pl -n=3 --type=word sequence.txt

          3-GRAMS (total count: 13)
          FIRST N-GRAM: A B C
          LAST N-GRAM: C F I
          -------------------------
          A B C   1
          B C H   1
          C F I   2
          C H I   1
          E F I   1
          F C F   1
          F I C   1
          F I E   1
          H I F   1
          I C F   1
          I E F   1
          I F C   1

          END OUTPUT BY Text::Ngrams
```

Order-2 Markov model

- Maximum likelihood estimator: max $p(a_{n+1}|a_n a_{n-1})$
- best performance in Order-k Markov models[2]

- extract last two elements in sequence history

```
$history: ...I H G H I H G H I F C F I H G H I F E F
$last_two = substr($history,-4,1) . ’ ’ . substr($history,-2,1);
# $last_two="E F"
```

[2] X. Yu, Y. Liu, D. Wei, M. Ting, “Hybrid Markov Models Used for Path Prediction,” International Conference on Computer Communications and Networks, 2006.

Order-2 Markov model

- find the largest ngram branch among branch with first two elements are `$last_two`

```
my %history_hash = $ng->get_ngrams;
my $prediction_count = 0;
my $prediction_key = 0;

foreach (keys (%history_hash)) {
    if ($_ =~ /^$last_two/) {
        if ($history_hash{$_} > $prediction_count) {
            $prediction_count = $history_hash{$_};
            $prediction_key = $_;
        }
    }
}
```

- prediction result

```
my $prediction_result;
print $prediction_result = substr($prediction_key,-1,1);
```

Multiple Inhabitant Case

- Using same framework!

```
$history: ...AB AC DF AC AE EG AI GG HI GG AH CG
$last_two = substr($last_two,-6,2) . ’ ’ . substr($last_two,-3,2);
# $last_two="AH CG"
```

Multiple Inhabitant Case

- find the largest ngram branch among branch with first two elements are `$last_two`

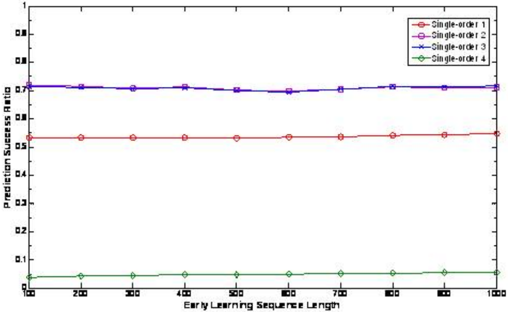
```
my %history_hash = $ng->get_ngrams;
my $prediction_count = 0; my $prediction_key = 0;

foreach (keys (%history_hash)) {
    if ($_ =~ /^$last_two/) {
        if ($history_hash{$_} > $prediction_count) {
            $prediction_count = $history_hash{$_};
            $prediction_key = $_;
        }
    }
}
```

- prediction result

```
my $prediction_result;
print $prediction_result = substr($prediction_key,-2,2);
```

Prediction Success Ratio



Conclusion

Summary

Improve location prediction algorithm

Sensor network testbed implementation

Location Prediction algorithm implementation

Perl is excellent!