Agenda

The lecture contains: A discussion on the information theoretic framework for tracking locations of mobile terminals on the basis of personal mobility patterns.
More specifically, the discussion is centered around Lezi-Update focusing on the following aspects of the framework, namely:

- Developing a personal mobility model.
- Applying information theoretic bound on personal mobility.
- Mathematical underpinning through Markov process.

Randomness in Personal Mobility

Personal and terminal mobility relationship

- Terminal mobility personal mobility closely tied.
- Sufficient knowledge on the past locations of an MT helps to make sense out of inherent randomness in mobility.
- Future locations can be predicted with high probability, even if updates lag behind in time.
- Approach, thus, leads to minimization of signal overheads incurred for both update and paging.
- The randomness is measured by entropy, so Shannon's formulation is applied to obtain an information theoretic bound on personal mobility.

Randomness in Personal Mobility

Measuring Information

- Information is measured in terms of number of bits.
- For example, 3000 bits are needed to transmit the results of 1000 rollings of a 8 sided (hypothetical) die in absence of any information.
- If the die is known to be biased, and the probability distribution is known, then a variable length coding can be used.
- The code should have prefix property ensuring that no code is a prefix of any code.

Randomness in Personal Mobility

Example

- For example, let the probability distribution for a biased die be:

$$p(i) = \begin{cases} 1/2^i, & \text{for } i \leq 7 \\ 1/2^{i-1}, & \text{for } i = 8, \end{cases}$$

- We use a variable length encoding which satisfies prefix property.
- The results of 1000 rollings of above biased die can be transmitted using only 1984 bits.
- So, with more information, the average number of bits required per result is reduced from 3 to 1.984.

Randomness in Personal Mobility

Example

- As another example, consider flipping of a biased coin.
- Let head show up just once in 1000 flips.
- Suppose the coin is tossed 1 million times.
- Let a 0 represent that the result of a toss is a head, and a 1 represents a tail.
- Without any clever encoding 1 bit will be required for result of each toss.
- But by transmitting the sequence number of the toss in which a head showed, the requirement reduces to $\log 10^6 = 20$ bits per 1000 flips.

Randomness in Personal Mobility

Length of Optimal Code

- Consider a conventional die with six faces to find an answer to above question.
- If the die is unbiased, the probability of occurrence of any value is $p = 1/6$, the number of bits required = $\log 6 = -\log(1/6) = -logp$.
- The result of one throw requires $\log 6 = 2.58$ bits:
  - Of course, the result a particular throw requires at least 3 bits.
  - However, by grouping g successive throws, the results can coded by less than 6^g bits.
- For a biased probability distribution, the number of bits required optimal code is determined by $\sum_x p(x) \times (-\log p(x))$.

Shannon's Entropy Formulation

Rare events and encoding

- Rare events have low probabilities and hence high information content.
- In the coin toss example, the information content in occurrence of a head is $-\log(1/1000) = \log 1000 = 9.9658$. So 10 bits are needed.
- But information contents in appearance of a tail is $-\log(999/1000) = 0.0044$ bit.
- So average information content is $(1/1000) \times 10 + (999/1000) \times 0.0044 = 0.0114$ bit

Shannon's Entropy Formulation

Self information in events

- If the probability of an outcome A_i of a random event A to be $p(A_i)$, then the expected value of self information of A is

$$H(A) = \sum_{i=1}^{n} p(A_i) \times \log\left(\frac{1}{p(A_i)}\right)$$
$$= -\sum_{i=1}^{n} p(A_i) \times \log p(A_i)$$

- According to Shannon $H(A)$ represents **entropy** of A.

Shannon's Entropy Formulation

Conditional entropy

- If X and Y are pair of discrete random variables with joint probability distribution $p(x, y)$, $x \in \mathcal{X}$, $y \in \mathcal{Y}$ then

$$H(X, Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log p(x, y)$$

- Conditional entropy $H(Y|X)$ is defined as:

$$H(Y|X) = \sum_{x \in \mathcal{X}} p(x) H(Y|X = x) = -\sum_{x \in \mathcal{X}} p(x) \sum_{y \in \mathcal{Y}} p(y|x) \log p(y|x)$$

$$= -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log p(x|y)$$

Shannon's Entropy Formulation

Joint entropy

$$
\begin{aligned}
H(X,Y) &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x,y) \\
&= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log(p(x)p(y|x)) \\
&= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x) \\
&\quad -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \\
&= -\sum_{x\in\mathcal{X}} p(x)\log p(x) + H(Y|X) \\
&= H(X) + H(Y|X)
\end{aligned}
$$

Personal Mobility as a Stochastic Process

Mobility as stochastic process

- A user's movement can be viewed as a random process.
- A user's movement can be captured as a sequence of cells.
- In GSM, each symbol represents a LAI consisting of several cells.
- Repetitions are possible and represents locality of movement.
- Characterizing mobility as probabilistic sequence, mobility can interpreted as a stochastic process.

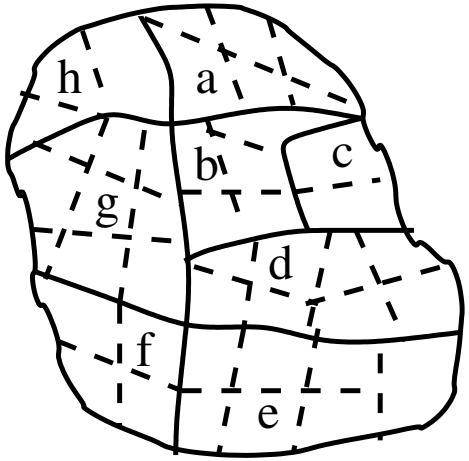
Personal Mobility as a Stochastic Process

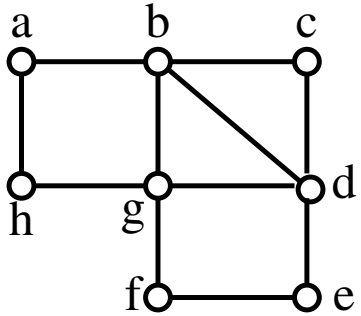
Graph modeling

At first we need to be clear on two issues:

1. How a user would move in a service area?
2. How movement can be recorded?

In a GSM type deployment group cells define a location area (LA).





Location Updates

Update types

- Updates can be distance based, time based, movement based or combinations thereof.
  - **Distance based**: by keeping track of Euclidean distance, mobile sends update, if distance travelled from last update crosses a threshold D.
  - **Movement based**: mobile sends an update if it has performed n cell crossings since the last update.
  - **Time based**: mobile sends periodic update.

Location Updates

Example

- Let service be started at 9.00AM. An example of zone crossings is provided in table below.

LAI Crossings until 5.00PM						
Time	11:04	11:32	11:57	3:18	4:12	4:52
Crossing	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow c$

LAI Crossings after 5.00PM				
Time	5:13	6:11	6:33	6:54
Crossing	$c \rightarrow d$	$d \rightarrow c$	$c \rightarrow b$	$b \rightarrow a$

Location Updates

Example

- With the movement history shown in previous slide, the zone sequences reported by various update schemes will be as shown below.

$T = 1$hr	*aaabbbbacdaaa* . . .
$T = 1/2$hr	*aaaaabbbbbbbbaabcddcaaaa* . . .
$M = 1$	*abababcdcba* . . .
$M = 2$	*aaacca* . . .
$T = 1$hr, $M = 1$	*aaababbbbbaabccddcbaaaa* . . .

Characterization of Mobility Model

Movement history

- The movement history is a string $v_1 v_2 v_3 \ldots$ of symbols from the alphabet ϑ, where v_i denotes the LAI reported in i th update.
- Mobility of a user is characterized as a stationary stochastic process $\mathcal{V} = \{V_i\}$ such that $V_i = v_i \in \vartheta$ if event i reported that user is in LAI v_i.

Characterization of Mobility Model

Movement histroy

- Joint distribution of any subsequence of V_i's is invariant with respect to shift in time axis.

$$Pr[V_1 = v_1, V_2 = v_2, \ldots V_n = v_n] = Pr[V_{l+1} = v_1, V_{l+2} = v_2, \ldots V_{n+l} = v_{n+l}]$$

  for every shift l and for all $v_i \in \vartheta$
- Movement history is a simple path or trajectory in $\mathcal{V}$.

Characterization of Mobility Model

Modeling movement history

- The above general model could aid in learning if and only a universal predictor can be constructed.
- Common models used for interpreting the movement history are:
  - Ignorant Model (IM)
  - Identically Independent Distribution (IID)
  - Markov Model (MM)
- IM disbelieves and disregards history. So all zone residence probabilities are same: 1/8 for each of the 8 zones for the running example.

Characterization of Mobility Model

IID Model

- Assumes that value random variables defining stochastic process are Identically and Independently Distributed.
- It uses relative frequencies of symbols as estimates of zone residence probabilities
- For the string: *aaababbbbbaabccddcbaaaa*, assuming time and movement based scheme,

$$p(a) = 10/23, p(b) = 8/23, p(c) = 3/23, p(d) = 2/23,$$
$$p(e) = p(f) = p(g) = p(h) = 0.$$

Markov Model

Markov chain

- Simplest MM assumes the stochastic process to be a time-invariant Markov chain defined by:

$$\begin{aligned} Pr[V_k = v_k | V_1 = v_1, \ldots, V_{k-1} = v_{k-1}] \\ = Pr[V_k = v_k | V_{k-1} = v_{k-1}] \\ = Pr[V_i = v_i | V_{i-1} = v_{i-1}] \end{aligned}$$

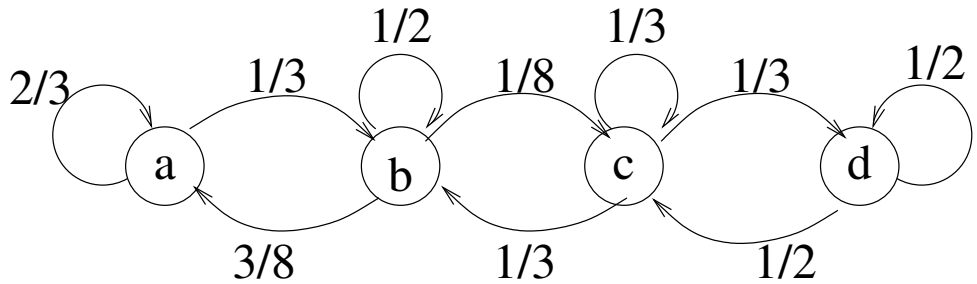
  for arbitrary choices for k and i.

- LAIs e, f, g, h not visited at all, and each acquire zero probability. The effective state space: $\{a, b, c, d\}$.

Markov Model

Markov chain

- One step transition probabilities are:
  $P_{i,j} = Pr[V_k = v_j | V_{k-1} = v_i]$, where $v_i, v_j \in \{a, b, c, d\}$, are estimated by relative counts.
- So, movement profile can be represented by:



Markov Model

Markov chain

- The corresponding transition probability matrix is:

$$P = \begin{bmatrix} 2/3 & 1/3 & 0 & 0 \\ 3/8 & 1/2 & 1/8 & 0 \\ 0 & 1/3 & 1/3 & 1/3 \\ 0 & 0 & 1/2 & 1/2 \end{bmatrix}$$

Markov Model

Adaptive modeling

- Let $\Pi = [p(a)\ p(b)\ p(c)\ p(d)]^T$ be steady state probability vector.
- Solving $\Pi = \Pi \times P$ with $p(a) + p(b) + p(c) + p(d) = 1$, we obtain $p(a) = 9/22$, $p(b) = 4/11$, $p(c) = 3/22$ and $p(d) = 1/11$.

Markov Model

Adaptability of models

- Ignorant model can never be adaptive.
- IID model is the first step toward adaptive modeling. paging strategy dependent on probabilities of $\{a, b, c, d, e, f, g, h\}$.
- Order-1 Markov model carries information to the extent of one symbol context.
- For uniformity IM can referred to as -1 order, and IID model as 0-order Markov models respectively.

Finite Context Models

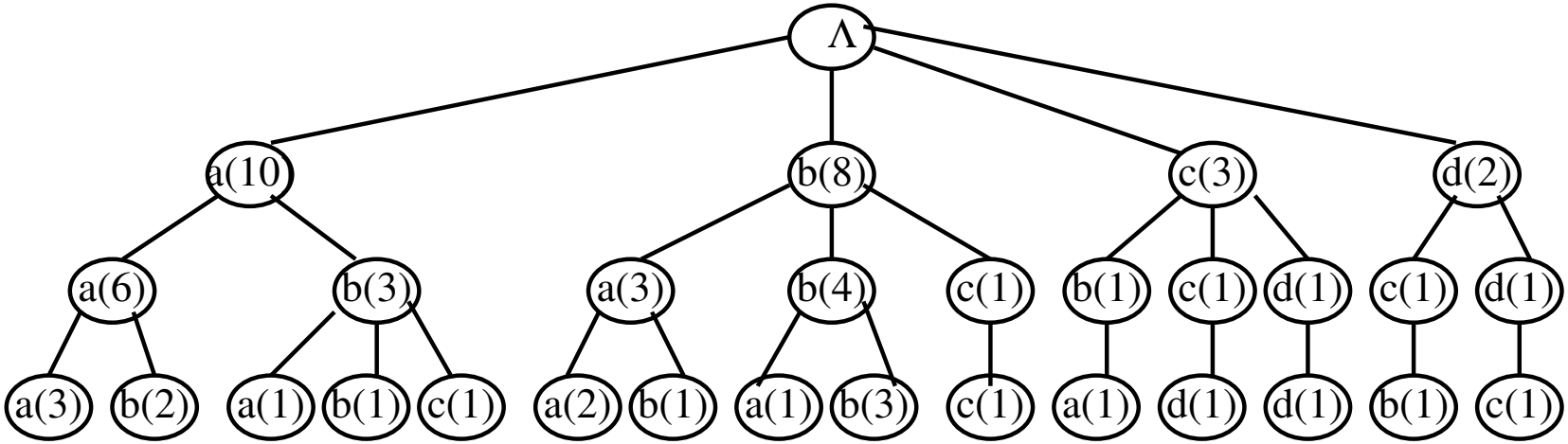
Higher order contexts

- Order-2 Markov model consisting of all order 2 contexts in the sequence *aaababbbbbaabccddcbaaaa* with their respective frequencies are:

Order-0	Order-1		Order-2		
$a(10)$	$a\|a(6)$	$b\|c(1)$	$a\|aa(3)$	$a\|ba(2)$	$a\|cb(1)$
$b(8)$	$b\|a(3)$	$c\|c(1)$	$b\|aa(2)$	$b\|ba(1)$	$d\|cc(1)$
$c(3)$	$a\|b(3)$	$d\|c(1)$	$a\|ab(1)$	$a\|bb(1)$	$d\|cd(1)$
$d(2)$	$b\|b(4)$	$c\|d(1)$	$b\|ab(1)$	$b\|bb(3)$	$b\|dc(1)$
	$c\|b(1)$	$d\|d(1)$	$c\|ab(1)$	$c\|bc(1)$	$c\|dd(1)$

Finite Context Models

Corresponding trie



Finite Context Models

Example of order-1 probability

- To compute steady state probability of the route *abcbcd* being taken
  - Relevant contexts are: $a|\Lambda$, $b|a$, $c|b$, $b|c$, $c|b$, $d|c$

$$=(9/22) \times (1/3) \times (1/8) \times (1/3) \times (1/8) \times (1/3)$$

$$=1/4224 = 2.37 \times 10^{-4}$$

Finite Context Models

Effects of higher order contexts

- Richness of information helps.
- Storing movement history for every movement is not practical.
- What order of context could lead to good prediction?
- Not that conditional is a decreasing function of the number of symbols in a stationary process.
- This implies that the advantage of higher order contexts dies out after a finite value.

Finite Context Models

Entropy of order-(-1) model

- V_is are independently and uniformly distributed, so $p(v) = 1/8$ for all $v \in \{a, b, c, d, e, f, g, h\}$ in the chosen running example.
- Due to independence $p(v_n|v_1, \ldots v_{n-1}) = p(v_n)$, therefore
  - $p(v_1, v_2, \ldots, v_n) = \{p(v_1)\}^n$
  - Both per symbol entropy $H(\mathcal{V})$ and conditional per symbol entropy $H'(\mathcal{V})$ are same.
- So, $H(\mathcal{V}) = H'(\mathcal{V}) = \log 8 = 3$ bits

Finite Context Models

Entropy of order-0 model

- V_is are Independently and Identically Distributed.
- Due to independence $p(v_n|v_1, \ldots v_{n-1}) = p(v_n)$, therefore

$$
\begin{aligned}
H(\mathcal{V}) = H'(\mathcal{V}) &= -\sum_{v} p(v) \log p(v) \\
&= (10/23) \times \log(23/10) + (8/23) \times \log(23/8) \\
&\quad + (3/23) \times \log(23/3) + (2/23) \times \log(23/2) \\
&\approx 1.742 \text{ bits}
\end{aligned}
$$

Finite Context Models

Entropy of order-1 model

- V_is form Markov chains. So $p(v_n|v_1 \ldots v_{n-1}) = p(v_n|v_{n-1}) = P_{v_{n-1},v_n}$.
- Substituting steady state probabilities $p(a) = 9/22$, $p(b) = 4/11$, $p(c) = 3/22$ and $p(d) = 1/11$.

$$H'(\mathcal{V}) = -\sum_v p(v) \left(\sum_j P_{i,j} \log P_{i,j} \right) = \frac{9}{22} \left(\frac{2}{3} \log \frac{3}{2} + \frac{1}{3} \log \frac{3}{1} \right)$$
$$+ \frac{4}{11} \left(\frac{3}{8} \log \frac{8}{3} + \frac{1}{2} \log \frac{2}{1} + \frac{1}{8} \log \frac{8}{1} \right)$$
$$+ \frac{3}{22} \left(3 \times \frac{1}{3} \log \frac{3}{1} \right) + \frac{1}{11} \left(2 \times \frac{1}{2} \log \frac{2}{1} \right) \approx 1.194 \text{ bits}$$

Trade-off between update and paging

- A mobile's location unknown for the duration since last update.
- The period of uncertainty is restricted to gap between two successive updates.
- Approach in LeZi update is to delay update if the path traversed by mobile is familiar.
- Information lag should not impact paging, because system can use prefix matching to predict location with high probability.

LZ algorithm

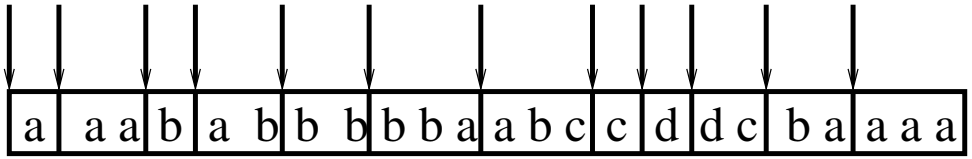
Encoding algorithm

- Lempel-Ziv algorithm for text compression has been used due to the requirement of having a fixed length coding scheme.

```
//  Encoder at mobile or compressing algorithm.
dictionary = null;
phrase w = null;
while (true) {
      wait for next symbol v;
      if (w.v in dictionary) w = w.v;
      else {
         encode <index(w), v>;
         add w.v to dictionary; w = null;
      }
}
```

LZ algorithm

Encoding example



#	entry	phrase
1	a	a
2	1+a	aa
3	b	b
4	1+b	ab
5	3+b	bb
6	5+a	bba
7	4+c	abc
8	c	c
9	d	d
10	9+c	dc
11	3+a	ba
12	2+a	aaa

Output:

0 a
1 a
0 b
1 b
3 b
5 a
4 c
0 c
0 d
9 c
3 a
2 a

LZ algorithm

Decoding algorithm

- Responsibility of decoding rests on the system side.

```
// Decoder at the system.
// It basically decompresses the string

while (true) {
   wait for the next code word <i, s>;
   decode phrase = dictionary[i].s;
   add phrase to dictinary;
   increment frequency of every prefix of the phrase;
}
```

LZ algorithm

Decoding example

input	#	entry	phrase
0 a	1	a	a
1 a	2	1+a	a a
0 b	3	b	b
1 b	4	1+b	a b
3 b	5	3+b	b b
5 a	6	5+a	b b a
4 c	7	4+c	a b c
0 c	8	c	c
0 d	9	d	d
9 c	10	9+c	d c
3 a	11	3+a	b a
2 a	12	2+a	a a a

a a a b a bb b b b aa b cc d d c b aa a a

LZ algorithm

Boundary cases

- Due to incremental parsing, repetition of a phrase will occur only at end.
- Encoding of a repeated phrase is: $< index, \lambda >$
- Decoding of input $< index, \lambda >$ outputs: $dictionary[index]$

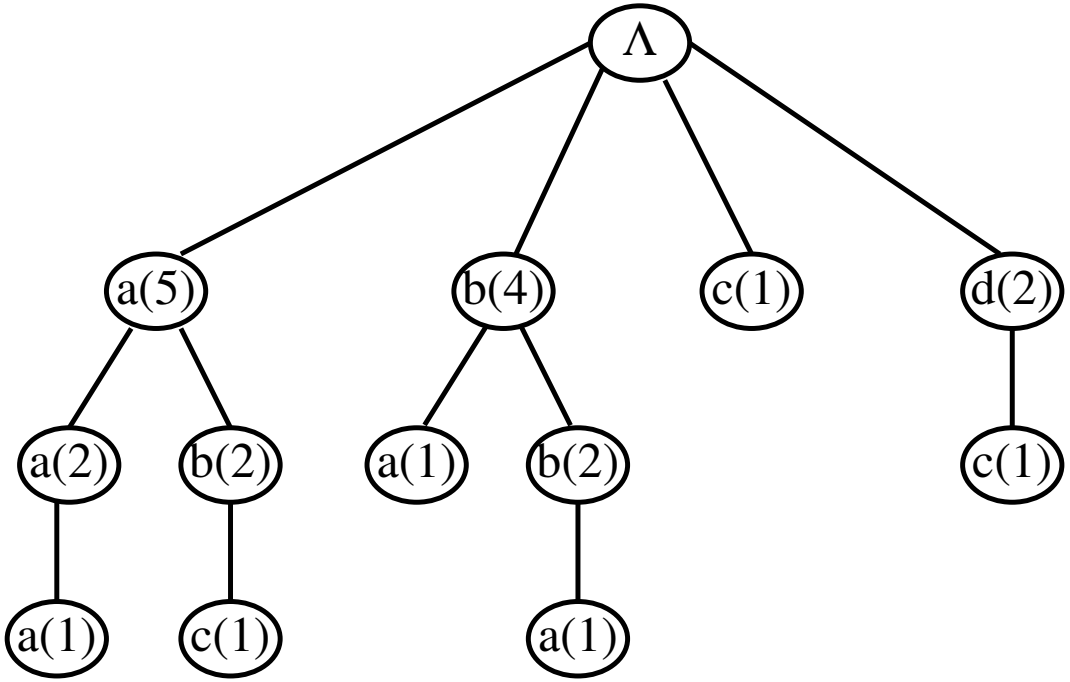
Incremental parsing

Storing dictionary

- Location updates is that the input string $v_1 v_2 \ldots v_n$ for $v_i \in \vartheta$ can be parsed into distinct substrings $w_1 w_2 \ldots w_k$ such that for all $j \geq 1$
  - Prefix of of substring w_j is a some w_i, $1 \leq i < j$.
- Due to prefix property substrings parsed can be stored in a **trie**.

Incremental parsing

Trie for classical LZ model



Incremental parsing

Convergence

- Larger and larger phrases are inserted into dictionary as incremental parsing progresses.
- Conditional probabilities for larger contexts start to build up.
- Predictability or richness of higher order Markov models also builds up.
- Therefore, the models based on Lempel-Ziv's compression eventually converge to a universal model

Incremental parsing

Problems with LZ model

- However, LZ compression based model fail to capture conditional entropy early on due to following reasons:
  - It works on one phrase at a time.
  - Decoding algorithm counts only the frequencies of prefixes of a decoded phrase.
  - It remains unaware about the contexts that straddle phrase boundaries.
  - All of the above slow down the rate of convergence.
- LeZi update uses an enhanced trie, where frequencies of of all prefixes of all suffixes are updated.
- By doing so, it captures the straddling effect.

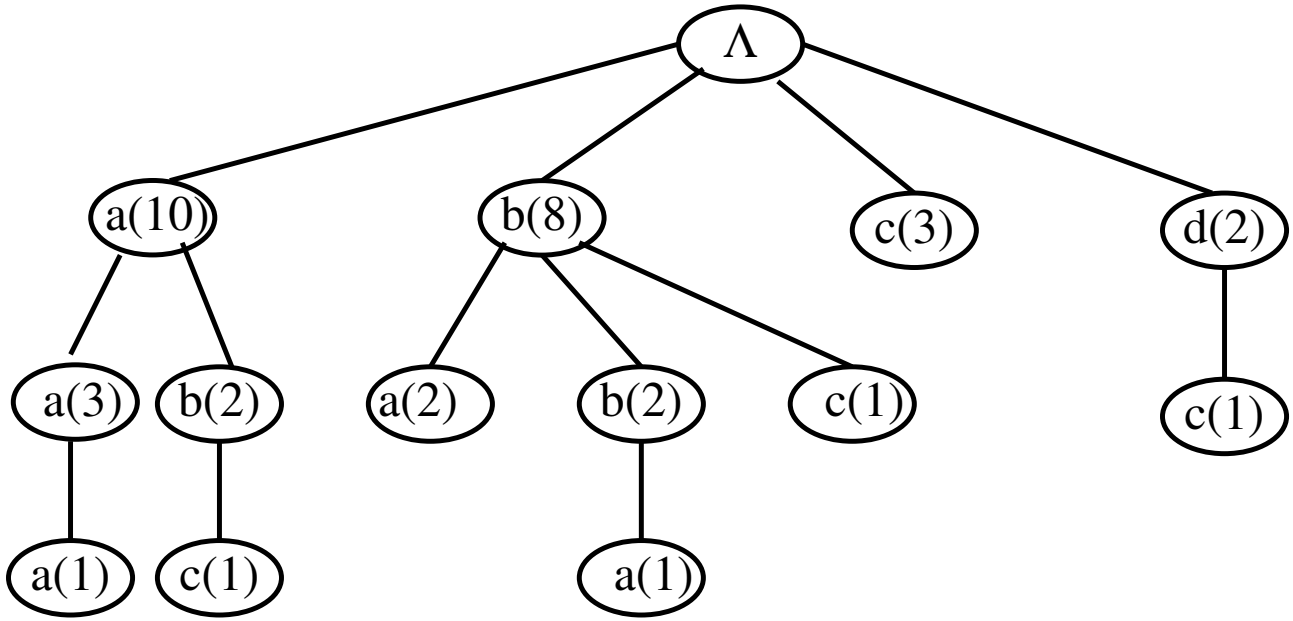
Incremental parsing

Frequency counting for LeZi Update

- Find all the prefix of all the suffixes of each incoming phrase.
- Increment the frequency each time a particular prefix is encountered starting from zero.
- Consider, phrase *aaa*:
  - Suffixes are *aaa*, *aa* and *a*, so frequency counts of all prefixes of *aaa*, *aa*, and *a* are incremented.
  - Frequency counts of *aa* incremented by 2.
  - Frequency counts of *a* incremented by 3.
- Total count for *a* can be found by considering phrases *a*, *aa*, *ab*, *ba*, *abc*, and *bba*.

Incremental parsing

Trie representation



Entropy Computation

Effect of using only prefixes

Conditional entropy without considering suffixes (see trie slide #39):

$$H(V_1) = \frac{5}{12}\log\frac{12}{5} + \frac{1}{3}\log 3 + \frac{1}{12}\log 12 + \frac{1}{6}\log 6$$
$$\approx 1.784. \text{ bits, and}$$

$$H(V_2|V_1) = \frac{5}{12}\left(2 \times \frac{1}{2}\log 2\right) + \frac{4}{12}\left(\frac{1}{3}\log 3 + \frac{2}{3}\log\frac{3}{2}\right)$$
$$\approx 0.723 \text{ bits (excludes terms contributing 0 to entropy).}$$

- Estimate for $H(\mathcal{V}) = (1.784 + 0.723)/2 = 1.254$ bits.
  Conditional probabilities of all order-2 contexts are 0.

Entropy Computation

Effect of using prefixes of suffixes

With enhanced tries (see slide #43), we still have

$$H(V_1) = \frac{10}{23}\log\frac{23}{10} + \frac{8}{23}\log\frac{23}{8} + \frac{3}{23}\log\frac{23}{3} + \frac{2}{23}\log\frac{23}{2}$$
$$\approx 1.742. \text{ bits, and}$$

$$H(V_2|V_1) = \frac{10}{23}\left(\frac{3}{5}\log\frac{5}{3} + \frac{2}{5}\log\frac{5}{2}\right) + \frac{8}{23}\left(2\times\frac{2}{5}\log\frac{5}{2} + \frac{1}{5}\log 5\right)$$
$$\approx 0.952 \text{ bits.}$$

- So the estimate for $H(\mathcal{V}) = (1.742 + 0.952)/2 = 1.347$ bits.
- Since entropy is more, it captures more information.

Probability assignment

- The prediction of a location for MN is guided by probability estimates of its possible locations.
- The underlying principle behind probability computation is PPM (prediction by partial matching).
- Our interest is in estimate of probability of occurrence of next symbol (zone) on path segment to be reported by next update.
- The segments are zones sequence generated when traversing from root to leaves of sub-tries representing current context.
- The estimated conditional probabilities for all zones at the current context gives the conditional probability distribution.

Probability assignment

- Suppose no LeZi type update received after receiving *aaa* and we want to find probability of predicting next symbol as *a*.
- Contexts that can be used are: *aa* and *a* and Λ.
- Possible paths that can be predicted with these contexts:

aa (Order-2)	a (Order-1)	Λ (Order-0)		
a | aa(1) Λ|aa(2)	a | a(2) aa | a(1) b | a(1) bc | a(1) Λ|a(5)	a(5) aa(2) ab(1) abc(1) b(3)	ba(2) bb(1) bba(1) bc(1) ba(a)	d(1) dc(1) Λ(1)

Probability assignment

- Start form highest order: context *aa*, the probability *a* is 1/3.
- Then fall back with probability 2/3 to order 1 (with null prediction), where the probability of *a*'s occurrence is $2/10 = 1/5$.
- Now fall back to order-0 (with null prediction), which has probability $5/10 = 1/2$.
- The probability of *a*'s occurrence in order-0 is 5/23.
- So blended probability of next symbol being *a* is

$$\frac{1}{3} + \frac{2}{3}\left(\frac{1}{5} + \frac{1}{2}\left(\frac{5}{23}\right)\right) = 0.5319$$

Probability assignment

- Consider occurrence of phrase *bba* after receiving *aaa*.
- *bba* does not occur in any of the contexts 1 or 2.
- Probability of escape from order-1 and order-2 contexts with null prediction is $(2/3) \times (1/2)$
  - for order 2 context (with phrase *aa*) probability of null prediction is 2/3 (out of 3 occurrence of *aa*.
  - for order 1 context (with phrase *a*) probability of null prediction is 1/2, because Λ occurs 5 times out of 10.
- The idea is to predict starting with some pre-determined highest order of context, and escape to lower orders until order 0 is reached.

Probability assignment

- Therefore, the blended probability of phrase *bba* is

$$0 + \frac{2}{3}\left(0 + \frac{1}{2}\left(\frac{1}{23}\right)\right) = 0.0145$$

- Now individual probabilities of symbols are: for *a* $(1/3)\times$ $0.0145 = .0048$ (since *a* occurs 1 once in phrase *bba*), for *b* it is $0.0145\times$ $(2/3) = .0097$.

Summary

- In this module we learned how terminal mobility can be managed by capturing personal mobility patterns.
- We studied how information theoretic bounds can be applied to terminal mobility based on the characterization of personal mobility patterns.
- The discussion mainly centered around LeZi update which was modeled after Lempel-Ziv text compression algorithm.
- LeZi update sends path based updates instead of zone based updates.
- It proposes a profile based paging strategy to overcome the gap between two successive location updates.