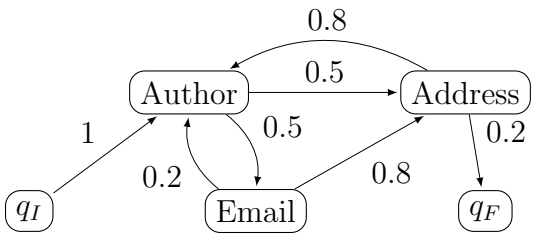
Organizing Web Information (CS 728) **Midterm Exam**
Computer Science and Engineering **2017-02-25 Saturday**
Indian Institute of Technology Bombay **17:30–19:30 SIC201**

NAME _____ ROLL _____

This exam has 10 printed page/s. Write your name and roll number on **EVERY SIDE (and not just sheet)**, because we may take apart your answer book and/or xerox it for correction. Write your answer clearly within the spaces provided and on any last blank page. Do not write inside the rectangles to be used for grading. **If you need more space than is provided, you probably made a mistake in interpreting the question.** Start with rough work elsewhere, but you need not attach rough work. Use the marks alongside each question for time management. **Illogical or incoherent answers are worse than wrong answers or even *no* answer, and may fetch negative credit.** You may not use any computing or communication device during the exam. You may use textbooks, class notes written by you, approved material downloaded **prior to the exam** from the course Web page, course news group, or the Internet, or notes made available by me for xeroxing. If you use class notes from other student/s, you must obtain them **prior to the exam** and **write down his/her/their name/s and roll number/s** here.

1. Consider the following simple Markov model with three nontrivial states:



1.a Write down an equivalent PCFG by completing the following. Note that only state transitions are shown, so there will be no production rules generating terminals.

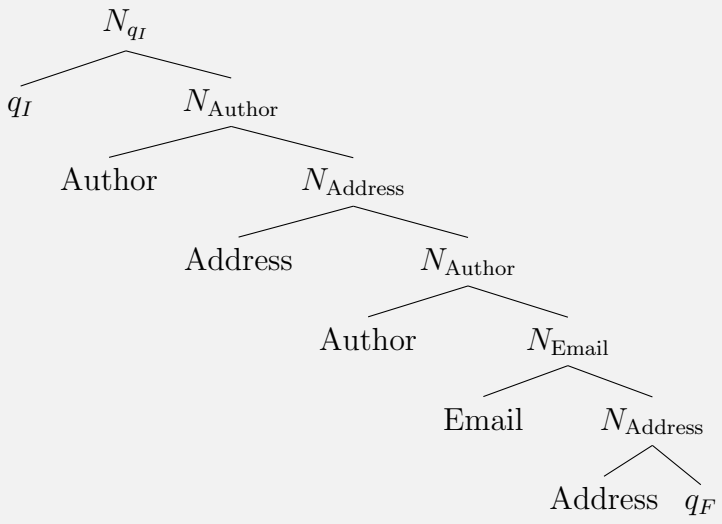
Probability	Production LHS	Production RHS
1	N_{q_I}	q_I N_{Author}
0.5	N_{Author}	Author N_{Address}
_____	N_{Author}	Author _____
_____	N_{Email}	_____
_____	N_{Email}	_____
_____	N_{Address}	_____
0.2	_____	Address q_F

		5

Probability	Production LHS	Production RHS
1	N_{q_I}	q_I N_{Author}
0.5	N_{Author}	Author N_{Address}
0.5	N_{Author}	Author N_{Email}
0.2	N_{Email}	Email N_{Author}
0.8	N_{Email}	Email N_{Address}
0.8	N_{Address}	Address N_{Author}
0.2	N_{Address}	Address q_F

1.b Draw the parse tree for the state sequence q_I, Author, Address, Author, Email, Address, q_F.

		3



1.c Write down the probability of the above parse tree. You need not simplify the expression.

		2

$1 \times 0.5 \times 0.8 \times 0.5 \times 0.8 \times 0.2$

2. Suppose we are designing a linear chain CRF to extract data from opening banners of research papers. Suppose such a banner is T tokens long, with tokens $x_1, \ldots, x_T$. States are numbered as in $[1, M]$, and say the state corresponding to the author field is $a \in [1, M]$. From domain knowledge we know that each paper must have at least one author. I.e., we need to enforce the constraint that state a occurs at least once, while we infer the most likely state sequence.

2.a As a first approach to enforce the constraint, we run the standard dynamic program (DP) to obtain the table $V(t, m)$ where $t \in [1, T-1]$ and $m \in [1, M]$. State 0 is the

designated start state.

$$V(0,*) = -\infty \qquad B(0,*) = \perp \qquad V(0,0) = 0$$
$$B(t,m) = \operatorname*{argmax}_{m'} V(t-1,m') + w \cdot \varphi(t,m',m)$$
$$V(t,m) = V(t-1,B(t,m)) + w \cdot \varphi(t,B(t,m),m)$$

where $B(t,m)$ is the backtrace table of best previous states. Then we compute the best second-last state $m^* = \operatorname{argmax}_{m'} V(T-1,m')$ and the sequence of states visited ending in m^*:

1: $S \leftarrow \varnothing$, $s \leftarrow m^*$
2: **for** $t = T-1, T-2, \ldots, 1$ **do**
3: $\quad S \leftarrow S \cup \{s\}$
4: $\quad s \leftarrow B(t,s)$
5: **return** S as $S(T-1,m^*)$

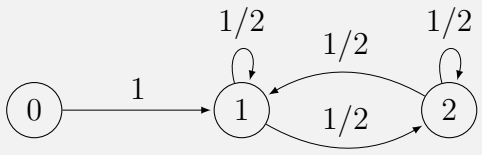
Finally, if $a \in S(T-1,m^*)$ already, we run a regular last time step for all $m \in [1,M]$:

$$V(T,m) = V(T-1,m^*) + w \cdot \varphi(T,m^*,m)$$

Otherwise, we force the last state at time T to be a.

Using a simple counterexample with one start state 0, two other states 1 and 2 and four symbols A, B, C, D, show that the above method is incorrect. (Hint: For simplicity, take transition probabilities out of the story by designing the state transitions suitably, and play with only the symbol emission probabilities.)

The state transitions are shown below; they are symmetric between states 1 and 2:



State	A	B	C	D
0	0	0	0	1
1	.3	.5	.2	0
2	0	.4	.1	.5

The symbol emission probabilities are shown above. Here $a = 2$. The observed symbol sequence is D, A, B, C. The optimal state sequence, subject to the constraint that state a must occur at least once, is (0, 1, 2, 1), with emission probability $.3 \times .4 \times .2 = .024$. The above algorithm will instead choose state sequence (0, 1, 1, 2) with emission probability $.3 \times .5 \times .1 = .015$. The first state (after 0) is forced to be 1, and for the first two steps, sequence 1,1 has probability $.3 \times .5 = .15$, compared to 1,2 with probability $.3 \times .4 = .12$.

2.b To enforce the constraint during inference, we run the standard dynamic program to obtain the table $V(t,m)$ where $t \in [1,T-1]$ and $m \in [1,M]$. We also write a method $S(t,m)$ that returns the set of states visited up to step t ending in state m, with the best score. For the last step, we compute

$$V(t,m) = \max_{m'} \begin{cases} V(t-1,m') + w \cdot \varphi(x_t,m',m), & a \in S(t-1,m') \\ V(t-1,m') + w \cdot \varphi(x_t,m',a), & a \notin S(t-1,m'), m = a \\ -\infty, & a \notin S(t-1,m'), m \neq a \end{cases}$$

Complete the base case and recursion for $S(t, m)$: $S(0, m) =$ _____ for $m \in [1, M]$, and for $m \in [1, M], t \in [1, T-1]$,

$$m^* = \underset{m'}{\text{argmax}} \quad \text{_____}$$
$$S(t, m) = S(t-1, m^*) \cup \text{_____}$$

		2

$S(0, m) = \varnothing$ (empty set) for $m \in [1, M]$, and for $m \in [1, M], t \in [1, T-1]$,

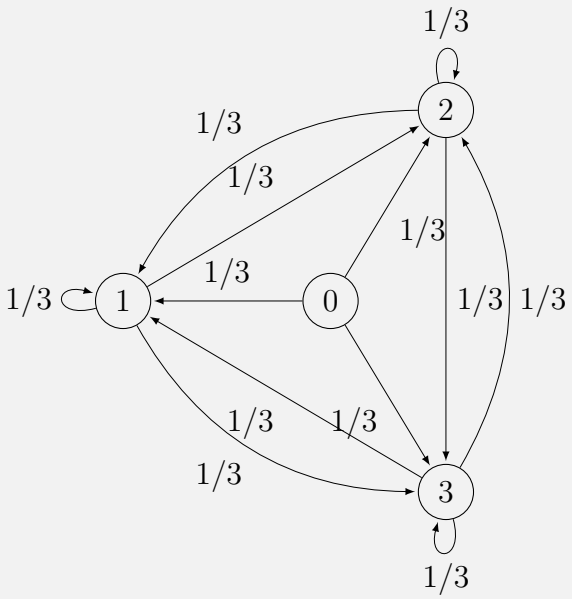
$$m^* = \underset{m'}{\text{argmax}} \begin{cases} V(t-1, m') + w \cdot \varphi(x_t, m', m), & a \in S(t-1, m') \\ V(t-1, m') + w \cdot \varphi(x_t, m', a), & a \notin S(t-1, m'), m = a \\ -\infty, & a \notin S(t-1, m'), m \neq a \end{cases}$$

$$S(t, m) = S(t-1, m^*) \cup m$$

2.c Using a simple counterexample with one start state 0 and three other states states 1, 2, 3, and four symbols A, B, C, D, show that the above method is incorrect. (Hint: For simplicity, take transition probabilities out of the story by designing the state transitions suitably, and play with only the symbol emission probabilities. Any correct counterexample, even with a different small number of states and symbols, will get full credit.)

		3

This less greedy/myopic approach will fix the earlier counterexample. However, we can deceive it easily.



State	A	B	C	D
0	0	0	0	1
1	.3	0	.4	.3
2	0	**1**	0	0
3	.2	0	.1	.7

Here $a = 3$, and the observed symbol sequence is D, A, B, C. The constrained optimal state sequence is $0, 3, 2, 1$. The less greedy algorithm looks at all

possible m' at $t-1$, so we render that useless by bottlenecking the sequence to pass through state 1, which is the only state that can emit B. The less greedy approach gets state sequence $0, 1, 2, 3$ with a strictly smaller objective.

It is possible to mislead both the "very greedy" and "slightly less greedy" algorithms with one start (0) and two other states (1 and 2). The transition and emission tables are given below.

Emission:

State	A	B	C	D
0	0	0	0	1
1	$.3+\epsilon$	.3	$.4-\epsilon$	0
2	.3	0	.3	.4

Transition:

State	0	1	2
0	0	.5	.5
1	0	.5	.5
2	0	.5	.5

The must-state is $a = 2$, and the observed symbol sequence is D, A, B, C as before. Observe that only state 1 can emit B. The constrained optimal sequence is $0, 2, 1, 1$ with score 0.0045. Both greedy versions choose $0, 1, 1, 2$ with score 0.003375.

2.d As an alternative approach, we can set $y_{\hat{t}} = a$ for $\hat{t} = 1, \ldots, T$ in turn, and find the best state assignments for other tokens. Complete the special case for $t = \hat{t}$:

$$V(\hat{t}, m) = \begin{cases} \max\limits_{m' \in [1,M]} \sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim & \text{if } m = \sim\sim\sim \\ \sim\sim\sim & \text{otherwise} \end{cases}$$

Cases $t \neq \hat{t}$ are unchanged. Write in two sentences why this is trivially guaranteed to be correct.

		3

$$V(\hat{t}, m) = \begin{cases} \max\limits_{m' \in [1,M]} V(t-1, m') + w \cdot \varphi(x_t, m', a) & \text{if } m = \underset{\sim}{a} \\ \underset{\sim\sim\sim}{-\infty} & \text{otherwise} \end{cases}$$

Any optimal sequence must have state a in *some* position, we call that $\hat{t}$ and try all possible values of $\hat{t}$.

2.e Write the overall pseudocode for the above approach and estimate the space and time needed.

		2

1: **for** $\hat{t} = 1, \ldots, T$ **do**
2: Solve above dynamic program to get $V(t, m|\hat{t})$
3: Best solution for $\hat{t}$ is $\max_m V(T, m|\hat{t})$
4: **return** $\max_{\hat{t}} \max_m V(T, m|\hat{t})$

The space required is TM. Each trial of $\hat{t}$ takes TM^2 time. So the total time is T^2M^2.

2.f Without the above restriction that state a must appear somewhere, the ordinary $V(t, m)$ table needs $O(TM)$ space. By increasing that only by a small constant

factor, can you solve the restricted problem, while substantially reducing overall time compared to the second approach in part 2.d above? If so, set up the new dynamic programming table and give full update expressions.

		3

Here we use the more conventional approach of keeping track of whether a has appeared in the state sequence or not, up to the current time. The dynamic programming table has two layers: $V(t, m, 0)$ (never seen a) and $V(t, m, 1_+)$ (seen a at least once).

$$
\begin{aligned}
V(0, *, 0) &= 0 \\
V(0, *, 1_+) &= -\infty \\
V(t, a, 0) &= -\infty && t \in [1, T] \\
V(t, m, 0) &= \max_{m' \neq a} V(t-1, m', 0) + w \cdot \varphi(x_t, m', m) && m \neq a, t \in [1, T] \\
V(t, a, 1_+) &= \max \begin{Bmatrix} \max_{m'} V(t-1, m', 0) \\ \max_{m'} V(t-1, m', 1) \end{Bmatrix} + w \cdot \varphi(x_t, m', a) && t \in [1, T] \\
V(t, m, 1_+) &= \max_{m'} V(t-1, m', 1_+) + w \cdot \varphi(x_t, m', m) && m \neq a, t \in [1, T]
\end{aligned}
$$

And finally we pick $\max_m V(T, m, 1_+)$. The space taken is $2TM$ and time is $O(TM^2)$, reduced from $O(T^2M^2)$ in part 2.d.

3. Consider fine-grained typing of entity mentions using context, where the types are organized in a tree hierarchy $\mathcal{T}$.

3.a As a preliminary exercise, suppose there are M items, each item u having a profit $s(u) \geq 0$. The overall score vector is denoted $\boldsymbol{s} = (s(u) : u \in [1, M])$. We have to select some subset of items, represented by vector $\boldsymbol{y} \in \{0, 1\}^M$. For the rest of this problem, we will relax to $\boldsymbol{y} \in [0, 1]^M$. Suppose we are given a total budget $K > 0$, which may not be an integer. Thus, we require $\|y\|_1 \leq K$ while seeking $\max_{\boldsymbol{y}} \boldsymbol{s} \cdot \boldsymbol{y}$. At most how many fractional elements need the optimal $\boldsymbol{y}$ contain? Justify.

		2

Suppose there are two fractional elements $y(i), y(j)$. If $s(i) \geq s(j)$, we can always set $y(i) \leftarrow y(i) + y(j)$ and $y(j) \leftarrow 0$ and not decrease the objective. Therefore there is an optimal $\boldsymbol{y}$ with at most one fractional element.

3.b In the fine-typing problem, any number of nodes $u \in \mathcal{T}$ may be valid choices for a specific mention, provided that once u is included, all its ancestors are included. A label $\boldsymbol{y}$ is called $\mathcal{T}$-closed if it satisfies this property. Let the root of $\mathcal{T}$ be labeled as node 0 and the parent of any other node u be denoted $p(u)$. Complete the following family of constraints:

$$\forall u \in \mathcal{T} \setminus \{0\} : \qquad y(\underbrace{\qquad\qquad}) \leq y(\underbrace{\qquad\qquad}).$$

		1

If $\boldsymbol{y} \in \{0,1\}^M$, we want to avoid the situation that $y(u) = 1$ but $y(p(u)) = 0$. Extending to the relaxed $\boldsymbol{y}$ vector,

$$\forall u \in \mathcal{T} \setminus \{0\} : \qquad y(u) \leq y(p(u)).$$

3.c Suppose a black box scoring algorithm outputs a score $s(u) \geq 0$ for each node, with the property that $s(u) \leq s(p(u))$. Will the approach of part 3.a give a $\mathcal{T}$-closed solution $\boldsymbol{y}$ automatically? If yes, argue informally; if not, modify the approach so that the optimal $\boldsymbol{y}$ is $\mathcal{T}$-closed. For simplicity you may assume there are no ties within $\boldsymbol{s}$.

		1

The same approach will work. Suppose $y(u) > y(p(u))$ for some u. However, by assumption, $s(u) < s(p(u))$. Let us replace $y(u)$ and $y(p(u))$ with their average, $\big(y(u) + y(p(u))\big)/2$. Clearly, the objective will strictly improve. But a child of u or the parent of $p(u)$ may cause a new violation. But we can keep fixing those. Because of no ties, the objective strictly increases, so we cannot run into cycles of updates.

3.d Consider a three-node $\mathcal{T}$ with root 0 and leaves 1 and 2. $s(0) = s(1) = 0$ and $s(2) = 1$. I.e., s is not $\mathcal{T}$-closed. For budget $K = 1$ what is the optimal $\boldsymbol{y}$? What is the optimal $\boldsymbol{y}$ for a general value of K?

		2

We seek to maximize $s(2)y(2) = y(2)$ subject to $y(0) + y(2) \leq 1$ and $y(0) \geq y(2)$. It is easy to see the solution is $y(0) = y(2) = 1/2$.
For general K, it is $\min\{1, K/2\}$.

3.e From the intuition gained from the above example, complete the following pseudocode for general s (not necessarily $\mathcal{T}$-closed).

Make each node u a singleton supernode S_u
Each S_u inherits $s(S_u) = s(u)$ of its node
Each supernode S has population $n(S) = 1$
The root supernode is assigned $y(S_0) = 1$
All other supernodes get $y(S_u) = 0$
$k \leftarrow 0$ /* Part of budget already allocated */
while $k < K$ **do**
    Find supernode S with largest $s(S)$ such that $y(S) = 0$
    if $y(p(S)) = 1$ **then**
        $y(S) \leftarrow \min\left\{1, \dfrac{K - \underset{\sim\sim\sim}{\sim\sim\sim}}{\underset{\sim\sim\sim}{\sim\sim\sim}}\right\}$
        $k \leftarrow k + \underset{\sim\sim\sim\sim\sim\sim\sim\sim\sim}{\sim\sim\sim\sim\sim\sim\sim\sim\sim}$
    else
        Condense S and $p(S)$ into S', reconnecting it suitably:
        $n(S') \leftarrow n(S) + n(p(S))$

15: $s(S') \leftarrow \frac{\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim\sim + s(p(S))n(p(S))}{n(S')}$
16: Set $y(S') \leftarrow 0$
17: **for** each supernode S **do**
18: Assign each node $u \in S$ the same $y(u)$ as $y(S)$

You do not have to formally prove that the algorithm is correct. | | | 5 |

1: Make each node u a singleton supernode S_u
2: Each S_u inherits $s(S_u) = s(u)$ of its node
3: Each supernode S has population $n(S) = 1$
4: The root supernode is assigned $y(S_0) = 1$
5: All other supernodes get $y(S_u) = 0$
6: $k \leftarrow 0$
7: **while** $k < K$ **do**
8: Find supernode S with largest $s(S)$ such that $y(S) = 0$
9: **if** $y(p(S)) = 1$ **then**
10: $y(S) \leftarrow \min\left\{1, \frac{K-k}{n(S)}\right\}$
11: $k \leftarrow k + n(S)$
12: **else**
13: Condense S and $p(S)$ into S', reconnecting it suitably:
14: $n(S') \leftarrow n(S) + n(p(S))$
15: $s(S') \leftarrow \frac{s(S)n(S) + s(p(S))n(p(S))}{n(S')}$
16: Set $y(S') \leftarrow 0$
17: **for** each supernode S **do**
18: Assign each node $u \in S$ the same $y(u)$ as $y(S)$

Proof of correctness can be found in `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=258128&tag=1`

4. In the DeepWalk algorithm, we embed each node u of an undirected graph $G = (V, E)$ as two vectors $r_u, c_u \in \mathbb{R}^K$. To do this, we first generate a sufficiently long random walk W on G. Let W_t be the tth node on the walk. Let T be a window size. We consider nodes W_i, W_j as "cooccurrent words" if $0 < |i - j| \leq T$, and feed this cooccurrent node pair into word2vec, which then produces embedding matrices $R, C \in \mathbb{R}^{|V| \times K}$.

4.a Let the transition matrix of G be denoted by A, where rows add up to 1. Let d_u be the degree of node u. $A_{uv} = 1/d_u$ if $(u, v) \in E$ and 0 otherwise. Let δ_u be a 1-hot vector with the 1 at position u. Complete: Ignoring end effects, each occurrence of node u in the random walk is recorded ~~~~~~~~~~ times by word2vec. | | | 1 |

The description above, taken from `https://www.ijcai.org/Proceedings/15/Papers/299.pdf`, is not quite equivalent to a pseudocode. Given

word2vec was mentioned, we can assume the standard Mikolov style pseudocode, with W indexed as $W_0, \ldots, W_{|W|-1}$.

1: **for** $f = T, T+1, \ldots, |W| - T - 1$ **do**
2: **for** $c = f - T, \ldots, f-1, f+1, \ldots f+T$ **do**
3: Report cooccurrence of W_f and W_c

When the occurence of u is picked up at position f, it is reported with $2T$ context words. For each such pair, there is also the converse pair where position c is focus and f is context. So overall the answer is $4T$, but we will give full credit for $2T$, given no pseudocode was provided.

4.b As a function of δ_u, A and t, the probability that a walk starting at u will visit node v at t steps later is the vth element of the vector _____. | | | 1 |

$\delta_u A^t$

4.c The expected number of times node v is visited within T steps of node u is the vth element of $\delta_u($_____$)$. | | | 2 |

Let I_t be the 0/1 indicator variable for event that v is visited exactly t steps after u. Then $\Pr(I_t = 1) = \mathbb{E}(I_t) = (\delta_u A^t)[v]$. By linearity, the expected number of times v is visited up to T steps after visiting u is $\sum_{t=1}^{T} \mathbb{E}(I_t) = \sum_{t=1}^{T} (\delta_u A^t)[v] = \big(\delta_u (A + A^2 + \cdots + A^T)\big)[v]$. (There was a spurious denominator T; people who ignored it and got the correct answer will get one bonus point.)

4.d If $N(u)$ is the number of times u appears in the cooccurrence data prepared for word2vec and $N(u, v)$ is the number of times the pair (u, v) appears in it, then it is known that skipgrams with softmax creates factors

$$R(u, *) \cdot C(v, *) \approx \log \frac{N(u, v)}{N(u)}.$$

Express the rhs above in terms of expressions derived in previous parts of this problem, using one or more of δ_u, d_u, A, T, v. | | | 2 |

Let $p(u, v, t) = (\delta_u A^t)[v]$. *After* u is visited, the expected fraction of the T steps that visit v is given by

$$\frac{\big(\delta_u (A + A^2 + \cdots + A^T)\big)[v]}{T} = \frac{1}{T} \sum_{t=1}^{T} p(u, v, t).$$

The situation for the T steps *before* u is visited may be less clear right away.

Let u be visited at time 0, and node v be visited at time $-t$.

$$\begin{aligned}\Pr(v_{@-t}|u_{@0}) &= \frac{\Pr(v_{@-t} \wedge u_{@0})}{\sum_{v'} \Pr(v'_{@-t} \wedge u_{@0})} = \frac{\Pr(v_{@-t}) \Pr(u_{@0}|v_{@-t})}{\sum_{v'} \Pr(v'_{@-t}) \Pr(u_{@0}|v'_{@-t})} \\ &= \frac{d_v p(v,u,t)}{\sum_{v'} d_{v'} p(v',u,t)} = \frac{d_u p(u,v,t)}{\sum_{v'} d_u p(u,v',t)} \\ &= \frac{p(u,v,t)}{\sum_{v'} p(u,v',t)} = p(u,v,t),\end{aligned} \tag{1}$$

because $\Pr(v_{@*}) \propto d_v$ in a connected undirected graph ("*" means any time), and, according to the balance equations of Markov walks, $d_v p(v,u,t) = d_u p(u,v,t)$. Thus, the before and after contexts of u have the same distribution over v's, and we can write:

$$\frac{N(u,v)}{N(u)} = \frac{1}{2T} \sum_{t=1}^{T} \Big[p(u,v,t) + \underline{p(u,v,t)} \Big] = \frac{1}{T} \sum_{t=1}^{T} p(u,v,t).$$

Full credit will be given if your answer matches up to the lhs of (1). (Thanks to Prof. Sundar Vishwanathan for clarifying the symmetry in the situations before and after u is visited.)

Total: 40