

Exercise 1 (Metropolis-Hastings Sampling of Independent Sets) 2+2+2=6 points

A set $I \subseteq V$ of nodes in a graph $G = (V, E)$ is called *independent*, if $(v, w) \notin E$ for all $v, w \in I$. Denote the independent sets of G by $\mathcal{I}_G \subseteq 2^{V(G)}$.

Fix some graph G . In this exercise, we want to uniformly sample from $\mathbb{U} := \mathcal{I}_G$ using Metropolis-Hastings sampling. Recall that the Metropolis-Hastings algorithm performs a random walk on an undirected, connected graph. To avoid confusion with G , we call this graph $\mathcal{H} = (V(\mathcal{H}), E(\mathcal{H}))$. (It is called \mathcal{G} in the lecture.)

- a) Define $V(\mathcal{H})$ and $E(\mathcal{H})$ for sampling from \mathcal{I}_G .

Hint: The edge relation $E(\mathcal{H})$ describes which elements of \mathcal{I}_G are "neighbours". Think of a useful notion of "being a neighbour" for sets of nodes of G .

- b) Find the maximum degree Δ of \mathcal{H} and show that \mathcal{H} is connected.
- c) Define transition probabilities $q_{I,J}$ for $I, J \in V(\mathcal{H})$. Show that the stationary distribution of the resulting Markov chain is the uniform distribution over \mathcal{I}_G .

Hint: Use slide 6.48 of the lecture and recall that $\mathcal{D} = Z\mathcal{P}$ where \mathcal{P} is the target distribution and Z is some constant. For our exercise, we know that \mathcal{P} is the uniform distribution on \mathcal{I}_G .

Solution: _____

- a) Define $V(\mathcal{H}) := \mathcal{I}_G$ and for every $I, J \in V(\mathcal{H})$ let $(I, J) \in E(\mathcal{H})$ if and only if $|(I \cup J) \setminus (I \cap J)| = 1$ (that is, I and J differ in exactly one node). Note that \mathcal{H} is undirected.

- b) Since the empty set \emptyset and all singletons $\{v\}$ are independent in G , $\emptyset \in V(\mathcal{H})$ with degree $n = |V(G)|$, so $\Delta \geq n$. As $\Delta \leq n$ by the definition of $E(\mathcal{H})$, $\Delta = n$.

Let $I, J \in V(\mathcal{H})$. Every subset of an independent set is independent. Thus, there exist paths from I to $I \cap J$ and from J to $I \cap J$ in \mathcal{H} (by excluding the extra nodes one by one). Therefore, I and J are connected and thus, \mathcal{H} is connected.

- c) Recall that for $I \in V(\mathcal{H})$, $N(I)$ denotes the neighbours of I , i. e. $\{J \in V(\mathcal{H}) : (I, J) \in E(\mathcal{H})\}$. We know that $\Delta = n$ and that \mathcal{D} is a scaled version of the uniform distribution on $V(\mathcal{H})$. In particular, $\mathcal{D}(I) = \mathcal{D}(J)$ for all $I, J \in V(\mathcal{H})$. Define transition probabilities according to slide 6.48:

$$q_{I,J} := \begin{cases} \frac{1}{n+1} & \text{if } (I, J) \in E(\mathcal{H}), \\ 1 - \sum_{K \in N(I)} q_{I,K} & \text{if } I = J, \\ 0 & \text{otherwise.} \end{cases}$$

By Theorem 6.22 of the lecture, the stationary distribution of the resulting Markov chain is the uniform distribution over \mathcal{I}_G .

Exercise 2 (Shuffling and Coupling)

0 points

Consider a Markov chain for shuffling n cards, where at each step a card is chosen uniformly at random and moved to the top. We run the chain until every card has been moved to the top at least once. Show that, at this stopping time, the state of the chain is uniformly distributed on all possible permutations of the cards.

Solution: _____

A corresponding Markov chain \mathcal{Q} for shuffling cards is given on slide 6.57.

We chose the coupling \mathcal{C} from slide 6.58 where at each step of \mathcal{C} we pick a random i and in both components move the card with **value** i to the top. Lemma 6.26 proves that this is a coupling of \mathcal{Q} .

Informally, the coupling Lemma (Lemma 6.24) states that "if the two marginal chains in the coupling \mathcal{C} have coupled in the sense that with high probability they are in the same state, then they must be close to the stationary distribution." We show that after having moved every card to the top at least once, it is guaranteed that both chains are in the same state ($\varepsilon = 0$). It is therefore guaranteed that we have reached the stationary distribution.

After moving the card with value i to the top, in all future states the position of i will be the same in both chains. We show this by induction on step t . Let $p_{(1,i)}^t$ be the position of i in chain 1 at step t . Let $p_{(2,i)}^t$ be the position of i in chain 2 at step t . Let t_0 be the first step at which i is moved to the top, clearly $p_{(1,i)}^{t_0} = p_{(2,i)}^{t_0} = 1$.

Now for $t > t_0$ we argue that $p_{(1,i)}^t = p_{(2,i)}^t$. We assume $p_{(1,i)}^{(t-1)} = p_{(2,i)}^{(t-1)}$.

For the position at t we look at 3 potential cases:

1. i is moved to the top, clearly then $p_{(1,i)}^t = p_{(2,i)}^t = 1$.
2. j is moved to the top and $p_{(1,j)}^{(t-1)} < p_{(1,i)}^{(t-1)}$ then j was moved to the top after the last time i was moved to the top and therefore $p_{(2,j)}^{(t-1)} < p_{(2,i)}^{(t-1)}$. In step t the position of i does not change in both chains. Formally $p_{(1,i)}^t = p_{(1,i)}^{(t-1)}$ and $p_{(2,i)}^t = p_{(2,i)}^{(t-1)}$ and we assume $p_{(1,i)}^{(t-1)} = p_{(2,i)}^{(t-1)}$, therefore $p_{(1,i)}^t = p_{(2,i)}^t$.
3. j is moved to the top and $p_{(1,j)}^{(t-1)} > p_{(1,i)}^{(t-1)}$ then j has not been moved to the top, after the last time i was moved to the top, so j can not be in front of i in chain 2 either, therefore $p_{(2,j)}^{(t-1)} > p_{(2,i)}^{(t-1)}$. As a result $p_{(1,i)}^t = p_{(1,i)}^{(t-1)} + 1$ and $p_{(2,i)}^t = p_{(2,i)}^{(t-1)} + 1$ and since we assume $p_{(1,i)}^{(t-1)} = p_{(2,i)}^{(t-1)}$ we also have $p_{(1,i)}^t = p_{(2,i)}^t$.

Therefore, after having moved the card for every value at least once, every value is at the same position in both chains and we know that the stationary distribution is reached. Lemma 6.25 states that the stationary distribution of \mathcal{Q} is the uniform distribution over all permutations of n cards.

Exercise 3 (Map-Reduce Algorithms for an Online Retailer) 2+2+1=5 points

We consider data records about the orders made at an online retailer. The data is given in key-value pairs of the shape $(o, (c, p, q, d))$ where

- o is an order ID,
- c is a customer ID,
- p is a product ID,
- q is a price in \$, and
- d is a timestamp.

A key value pair $(o, (c, p, q, d))$ represents an order made by customer c on date (and time) d , having ordered product p for a price of q dollars. If an order consists of multiple product purchases, then it is captured by multiple pairs with the same order ID (and same customer ID and time stamp). We assume that the price q of any product p stays the same over all orders.

Solve the following tasks.

- a) *Product Revenue*. Specify a *single round* Map-Reduce algorithm that outputs all key-value pairs (p, s) where p is a product, and s is the total income generated for the retailer by selling p (that is, the sum of p 's price over all orders of p).

Hint: Keep in mind that a product may be ordered multiple times by the same customer, possibly even in the same order.

- b) *Bargain Shoppers*. Specify a *single round* Map-Reduce algorithm that outputs all key-value pairs $(c, (p, q))$, where c is a customer who *only* bought products with price $< 20\$$, and p is a product they bought with price q . You do not need to handle duplicates in this task.
- c) *Also Bought*. Let p_0 be a fixed product ID. The Map-Reduce algorithm shown below outputs all key-value pairs (c, p) , where c is a customer *who ordered* p_0 and p is another product they bought.

MAP: On input $(o, (c, p, q, d))$ emit $(p_0, (c, p))$.

REDUCE: On input (p_0, values) do
for all c appearing in a pair in values :
if $(c, p_0) \in \text{values}$, emit (c, p) for all $(c, p) \in \text{values}$ with $p \neq p_0$.

This algorithm technically does the job, but it is a terrible Map-Reduce algorithm. Explain why.

Remark: Use the „on input ..., [do some computation,] emit ...“ format for specifying your pseudocode. Make sure that your algorithms are not unnecessarily inefficient.

Solution: _____

- a)** MAP: On input $(o, (c, p, q, d))$, emit (p, q) .
REDUCE: On input (p, values) , emit (p, s) where $s = |\text{values}| \cdot q$ (all entries of values are equal to q).
- b)** MAP: On input $(o, (c, p, q, d))$, emit $(c, (p, q))$.
REDUCE: On input (c, values) , if values contains no tuples (p, q) with $q \geq 20$, then emit $(c, (p, q))$ for all $(p, q) \in \text{values}$.
- c)** The algorithm only emits pairs with a single key in the map phase, so there is only one reduce task. The whole computation will happen on a single node.

Exercise 4 (Analysis of Matrix Multiplication)

6+3=9 points

In this exercise, we consider another measure for analysing Map-Reduce algorithms, and apply it to the algorithms for matrix multiplication that were shown in the lecture : the single-round algorithm \mathcal{A}_1 (slide 7.34), the two-round algorithm \mathcal{A}_2 (slide 7.33), and the generalized single-round algorithm \mathcal{A}_1^s (slide 7.46, where s is the „number of stripes“ parameter).

As input, we consider two matrices $A, B \in \mathbb{R}^{n \times n}$. For the three algorithms presented in the lecture solve the following tasks, ignoring the possibility of node failures.

- a) For \mathcal{A}_1 , \mathcal{A}_2 and \mathcal{A}_1^s , and for each phase of the respective algorithm determine
- the total number of tasks, and
 - the maximum running time of a single task

Assume the worst case input and give your result depending on n (and for \mathcal{A}_1^s additionally depending on s).

Remark: For the maximum running time part, use \mathcal{O} -notation and just count the number of operations.

		Total number of tasks	Running time of a single task
\mathcal{A}_1	MAP: REDUCE:		
\mathcal{A}_2	First MAP: First REDUCE: Second MAP: Second REDUCE:		
\mathcal{A}_1^s	MAP: REDUCE:		

- b) Now, for all three algorithms, compute the worst case running time (in \mathcal{O} -notation) using the measure above, **taking parallelization over N nodes into account**, for

	$N \in \Theta(1)$ nodes	$N \in \Theta(n)$ nodes	$N \in \Theta(n^2)$ nodes:
\mathcal{A}_1			
\mathcal{A}_2			
\mathcal{A}_1^s			

Solution: _____

a)

		Total number of tasks	Running time of a single task
\mathcal{A}_1	MAP: REDUCE:	$2n^2$ n^2	$\mathcal{O}(n)$ $\mathcal{O}(n)$
\mathcal{A}_2	First MAP: First REDUCE: Second MAP: Second REDUCE:	$2n^2$ n n^3 n^2	$\mathcal{O}(1)$ $\mathcal{O}(n^2)$ $\mathcal{O}(1)$ $\mathcal{O}(n)$
\mathcal{A}_1^s	MAP: REDUCE:	$2n^2$ s^2	$\mathcal{O}(s)$ $\mathcal{O}\left(\left(\frac{n}{s}\right)^2 \cdot n\right)$

b)

	$N \in \Theta(1)$ nodes	$N \in \Theta(n)$ nodes	$N \in \Theta(n^2)$ nodes:
\mathcal{A}_1	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$
\mathcal{A}_2	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
\mathcal{A}_1^s	$\mathcal{O}(2n^2s + s^2\left(\frac{n}{2}\right)^2n) = \mathcal{O}(n^3)$	$\mathcal{O}(2ns + \lceil \frac{s^2}{n} \rceil \left(\frac{n}{s}\right)^2n)$	$\mathcal{O}(s + \lceil \frac{s^2}{n^2} \rceil \cdot \frac{n^2}{s^2} \cdot n)$