3.5)

Given to compute $P(Y|X=0)$, $P(Y|X=1)$,

$P(Z_1=1|Y)$, $P(Z_2=1|Y)$

$$P(Y|X=0) = P(Y_1, Y_2 Y_3 | X=0)$$

$$= P(Y_1|X=0)\, P(Y_2 Y_3 | X=0, Y_1)$$ [Product rule with evidence]

Now from rule ② of d-separation $Y_2, Y_3$ are conditionally independent of $Y_1$ given $X$:

Hence

$$= P(Y_1|X=0)\, P(Y_2, Y_3 | X=0)$$

$$= P(Y_1|X=0)\, P(Y_2|X=0)\, P(Y_3|X=0, Y_2)$$ [Product rule with evidence].

Again from rule ② of d-separation,

$Y_2, Y_3$ are conditionally independent given $X$.

$$= P(Y_1|X=0)\, P(Y_2|X=0)\, P(Y_3|X=0) \quad -①$$

Similarly

$$P(Y|X=1) = P(Y_1|X=1)\, P(Y_2|X=1)\, P(Y_3|X=1) \quad -②$$

$P(Z_1=1|Y) =$ $P(Z_1=1|Y_1 Y_2 Y_3) \Rightarrow$ CPT table $-③$

$P(Z_2=1|Y) =$ $P(Z_2=1|Y_1 Y_2 Y_3) \Rightarrow$ CPT table. $-④$

Computing values first two rows of the table.

First row

| $Y_1$ | $Y_2$ | $Y_3$ | $Y$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |

$$P(Y|X=0) = P(Y_1|X=0)\,P(Y_2|X=0)\,P(Y_3|X=0) \quad [\text{from eq } ①]$$

$$= P(Y_1=0|X=0)\,P(Y_2=0|X=0)\,P(Y_3=0|X=0)$$

$$= (1-0.75)(1-0.5)(1-0.25)$$

$$= 0.25 \times 0.5 \times 0.75$$

$$= 0.09375$$

$$P(Y|X=1) = P(Y_1=0|X=1)\,P(Y_2=0|X=1)\,P(Y_3=0|X=1) \quad [\text{from eq } ②]$$

$$= (1-0.5)(1-0.25)(1-0.75)$$

$$= 0.5 \times 0.75 \times 0.25$$

$$= 0.09375$$

$P(Z_1=1 \mid Y_1=0, Y_2=0, Y_3=0)$ = 0.9 (from CPT table)

$P(Z_2=1 \mid Y_1=0, Y_2=0, Y_3=0)$ = 0.1 (from CPT table)

Second row

| $Y_1$ | $Y_2$ | $Y_3$ | $Y$ |
|---|---|---|---|
| 1 | 0 | 0 | 2 |

$$P(Y|X=0) = P(Y_1=1|X=0)\,P(Y_2=0|X=0)\,P(Y_3=0|X=0)$$

$$= 0.75\,(1-0.5)(1-0.25)$$

$$= 0.75 \times 0.5 \times 0.75$$

$$= 0.28125$$

$$P(Y|X=1) = P(Y_1=1|X=1)\,P(Y_2=0|X=1)\,P(Y_3=0|X=1)$$

$$= 0.5 \times (1-0.25) \times (1-0.75)$$

$$= 0.5 \times 0.75 \times 0.25$$

$$= 0.09375$$

$P(Z_1=1 | Y_1=1, Y_2=0, Y_3=0) = 0.8$ (CPT table).

$P(Z_2=1 | Y_1=1, Y_2=0, Y_3=0) = 0.2$ (CPT table).

Table

| $Y_1$ | $Y_2$ | $Y_3$ | $Y$ | $P(Y\|X=0)$ | $P(Y\|X=1)$ | $P(Z_1=1\|Y)$ | $P(Z_2=1\|Y)$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0.09375 | 0.09375 | 0.9 | 0.1 |
| 1 | 0 | 0 | 2 | 0.28125 | 0.09375 | 0.8 | 0.2 |
| 0 | 1 | 0 | 3 | 0.09375 | 0.03125 | 0.7 | 0.3 |
| 0 | 0 | 1 | 4 | 0.03125 | 0.28125 | 0.6 | 0.4 |
| 1 | 1 | 0 | 5 | 0.28125 | 0.03125 | 0.5 | 0.5 |
| 1 | 0 | 1 | 6 | 0.09375 | 0.28125 | 0.4 | 0.6 |
| 0 | 1 | 1 | 7 | 0.03125 | 0.09375 | 0.3 | 0.7 |
| 1 | 1 | 1 | 8 | 0.09375 | 0.09375 | 0.2 | 0.8 |

3.6)

a) To show

$$\sum_{z} P(Z=z \mid B_1 B_2 \ldots B_n) = 1 \quad \text{for sum of } z \text{ over } [-\alpha, \alpha]$$

$$= \sum_{z=-\alpha}^{\alpha} P(Z=z \mid B_1 B_2 \ldots B_n)$$

$$= \sum_{z=-\alpha}^{\alpha} \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{|z-f(B)|}$$

if $z$ varies from $-\infty$ to $\infty$ then $z-f(B)$ will also vary from $-\infty$ to $\infty$, hence we can replace $z$ with $y$ where $y = z-f(B)$

$$= \sum_{y=-\infty}^{\alpha} \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{|y|}$$

$$= \sum_{y=-\infty}^{-1} \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{|y|} + \sum_{y=1}^{\infty} \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{|y|} + \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{(0)} \quad \text{L} [y=0]$$

$$= \frac{1-\alpha}{1+\alpha} + \underbrace{\sum_{y=-\alpha}^{-1} \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{|y|}}_{①} + \underbrace{\sum_{y=1}^{\infty} \left(\frac{1-\alpha}{1+\alpha}\right) \alpha^{|y|}}_{②}$$

Now ① & ② are the same because in both the cases $|y|$ will vary from 1 to $\infty$.

Hence we get

$$= \frac{1-\alpha}{1+\alpha} + 2\sum_{y=1}^{\infty}\left(\frac{1-\alpha}{1+\alpha}\right)\alpha^y$$

$$= \frac{1-\alpha}{1+\alpha} + 2\left(\frac{1-\alpha}{1+\alpha}\right)\left[\sum_{y=1}^{\infty}\alpha^y\right] \Rightarrow \text{Infinite GP}$$

$$= \frac{1-\alpha}{1+\alpha} + 2\left(\frac{1-\alpha}{1+\alpha}\right)\left(\frac{\alpha}{1-\alpha}\right)$$

$$= \frac{1-\alpha}{1+\alpha} + \frac{2\alpha}{1+\alpha}$$

$$= \frac{1+\alpha}{1+\alpha} = 1.$$

Hence $P(Z = z/B_1 B_2 \cdots B_n) = 1$ where $z \in [-\infty, \infty]$.

$z$.

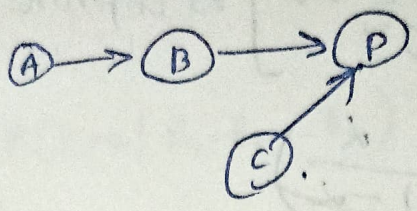
c) I have plotted the values of probability in logspace for 1 million experiments. for $P(B_i = 1 / Z = 128)$ for $i \in \{2, 5, 8, 10\}$.

I have also print the last 15 probability values for each $i \in \{2, 5, 8, 10\}$, we can clearly see that the most significant bits are the same in all the values, and also the plots appear to be converged. Hence we can confident say that estimates have converged to good degree of precision.

3.7)

a) To compute

$$P(B|A,C,D)$$




$$P(\underset{X}{B}|\underset{E}{A,C,D})$$

$$\left[ P(X|Y,E) = \frac{P(Y|X,E)\,P(X|E)}{P(Y|E)} \right]$$

[Bayes rule with evidence]

$$P\,\frac{(A|B,C,D)\,P(B|C,D)}{P(A|C,D)}$$

$$= \frac{P(D|B,A,C)\,P(B|A,C)}{P(D|A,C)}$$

$$= \frac{P(D|B,C)\,P(B|A)}{P(D|A,C)}$$

$$\left[ \begin{array}{l} P(D|B,C,A) = P(D|B,C) \\ \text{using d-separation rule (1)} \\ P(B|A,C) = P(B|A) \\ \text{Using d-separation rule (3)} \end{array} \right]$$

$$= \frac{P(D|B,C)\,P(B|A)}{\sum_b P(D,B=b|A,C)}$$

$$= \frac{P(D|B,C)\,P(B|A)}{\sum_b P(B=b|A,C)\,P(D|B=b,A,C)}$$

[Product rule with evidence]

$$= \frac{P(D|B,C)\,P(B|A)}{\sum_b P(B=b|A)\,P(D|B=b,C)}$$

$P(B=b|A,C) = P(B=b|A)$ using d-separation rule (3)

$P(D|B=b,C) = P(D|B=b,A,C)$ using d-separation rule (1)

$$= \frac{P(D|B,C)\,P(B|A)}{\sum_b P(B=b|A)\,P(D|B=b,C)} \quad \text{- eq (1)}$$

we can compute $P(B|A)$ $P(D|B,C)$ from CPT tables.

b) $P(B \mid \{A,C,D\}, \{E,F\})$

B is conditionally independent of $\{E,F\}$ given $\{A,C,D\}$.

All paths from node E to B is blocked by node 'C' d-separation rule (2).

All paths from node F to B is blocked by node 'A', d-separation rule (2).

Hence $P(B|A,C,D,E,F) = P(B|A,C,D)$

$$= \frac{P(D|B,C)P(B|A)}{\sum_b P(B=b|A)\,P(D|B=b,C)}$$

c) To compute:

$P(B, E, F \mid A, C, D)$

$= P(B \mid A, C, D)\, P(E, F \mid A, C, D, B)$ [Product rule with evidence]

$= P(B \mid A, C, D)\, P(E \mid A, B, C, D)\, P(F \mid A, C, D, B, E)$ [ " ]

$= P(B \mid A, C, D)\, P(E \mid A, B, C, D)\, P(F \mid A, B, C, D, E)$ [ " ]

$P(B \mid A, C, D)$ has already been computed in (i) [eqn ①]

$P(E \mid A, B, C, D) = P(E \mid C) \Rightarrow$ CPT

using d-separation rule ②

A, B, D are blocked by C.

$P(F \mid A, B, C, D, E) = P(F \mid A) \Rightarrow$ CPT

Using d-separation rule ②

B, C, D, E are blocked by A.

$= P(B \mid A, C, D)\, P(E \mid C)\, P(F \mid A)$.

3.8) a) Given 'T' samples $\{q_t, x_t, y_t, z_t\}_{t=1}^{T}$

We need to estimate $P(Q = q \mid E = e)$

From the formula of likelihood weighing we have

$$P(Q=q|E=e) = \frac{\sum_{t=1}^{T} I(q,q_t)\, P(E=e|q_t,x_t,y_t,z_t)}{\sum_{t=1}^{T} P(E=e|q_t,x_t,y_t,z_t)}$$

Now $P(E|X,Q,Y,Z) = P(E|Y,Z)$

All paths from $X,Q$ are blocked by $Y,Z$ using d-separation rule ①.

$$= \frac{\sum_{t=1}^{T} I(q,q_t)\cdot P(E=e|y_t,z_t)}{\sum_{t=1}^{T} P(E=e|y_t,z_t).}$$

we can estimate $P(E=e|y_t,z_t)$ from CPT table.

b) Given 'T' samples $\{q_{1t}, q_{2t}, x_t, y_t, z_t\}_{t=1}^{T}$

To estimate $P(Q_1=q_1|E_1=e_1, E_2=e_2)^{Q_2=q_2}$, similar to bit a) we have

~~$P(Q=q_1, E=e_1)$~~

$$P(Q_1=q_1, Q_2=q_2|E_1=e_1, E_2=e_2)$$

$$= \frac{\sum_{t=1}^{T} I(q_1,q_{1t})\, I(q_2,q_{2t})\, P(E_1=e_1, E_2=e_2|q_{1t},q_{2t},x_t,y_t,z_t)}{\sum_{t=1}^{T} P(E_1=e_1, E_2=e_2|q_{1t},q_{2t},x_t,y_t,z_t)}$$

Now

$$P(E_1 = e_1, E_2 = e_2 \mid q_{1t}, q_{2t}, x_t, y_t, z_t)$$

$$= P(E_1 = e_1 \mid q_{1t}, q_{2t}, x_t, y_t, z_t)\; P(E_2 = e_2 \mid q_{1t}, q_{2t}, x_t, y_t, z_t, E_1 = e_1)$$

$E_1$ is conditionally independent of $q_2, z, Y$ given $q_1, X$ using d-separation rules.

Also $E_2$ is conditionally independent of $q_2, x, y, q_1$ given $E_1, z$

Hence we get

$$P(q_1 = q_1, q_2 = q_2 \mid E_1 = e_1, E_2 = e_2)$$

$$= \sum_{t=1}^{T} I(q_1, q_{1t})\, I(q_2, q_{2t}) \cdot P(E_1 = e_1 \mid q_1 = q_{1t}, X = x_t)\; P(E_2 = e_2 \mid E_1 = e_1, z = z_t)$$

---

$$P(E_1 = e_1 \mid q_1 = q_{1t}, X = x_t)\; P(E_2 = e_2 \mid E_1 = e_1, z = z_t)$$

[ values of $P(E_1 = e_1 \mid q_1 = q_{1t}, X = x_t)$ and $P(E_2 = e_2 \mid E_1 = e_1, z = z_t)$ can be computed from the CPT table ].

3.1)

a) To compute

$P(X_{t+1} = j \mid X_1 = i) = [A^t]_{ij}$ where $A^t$ is the power $t^{th}$ of matrix A.

We will prove this using induction,

⇒ for $t = 1$

$P(X_2 = j \mid X_1 = i) = [A]_{ij}$ which is true

as given $A_{ij} = P(X_{t+1} = j \mid X_t = i)$

$\Rightarrow P(X_2 = j \mid X_1 = i) = A_{ij}$

⇒ Assume true for $t = K$ to be true.

ie.,

$P(X_{t+1} = j \mid X_1 = i) = [A^t]_{ij}$

⇒ Now to prove $K+1$ to be true that is $t+1$ to be true

$P(X_{t+1+1} = j \mid X_1 = i) \Rightarrow [A^{t+1}]_{ij}$

LHS $P(X_{t+2} = j \mid X_1 = i)$

$= \sum_{r=1}^{m} P(X_{t+2} = j, X_{t+1} = r \mid X_1 = i)$ [Marginalization $X_t \in \{1, 2 \ldots m\}$ given]

$= \sum_{r=1}^{m} P(X_{t+1} = r \mid X_1 = i)\, P(X_{t+2} = j \mid X_1 = i, X_{t+1} = r)$ [product rule].

$$= \sum_{r=1}^{m} P(X_{t+1}=r \mid X_1=i)\; P(X_{t+2}=j \mid X_{t+1}=r)$$

$X_{t+2}$ is conditionally independent of $X_1$ given $X_{t+1}$ [d-separation rule ①].

$$= \sum_{r=1}^{m} [A^t]_{ir} \cdot P(X_{t+2}=j \mid X_{t+1}=r)$$

$[A^t]_{ir}$ → from induction step

$$= \sum_{r=1}^{m} [A^t]_{ir}\, [A]_{rj} \quad \rightarrow \text{eq ②}$$

$[A]_{rj}$ → given.

basically eq ②, we are multiplying $i^{th}$ row of $A^t$ with $j^{th}$ column of $A$, by defination of matrix multiplication we know that

$$[AB]_{ij} = \sum_{r=1}^{K} [A]_{ir}\, [B]_{rj}$$

where $A \Rightarrow x \times K$

$B \Rightarrow K \times y$

Hence:

eq ② implies $\sum_{r=1}^{m} [A^t]_{ir}\, [A]_{rj} = [A^{t+1}]_{ij}$

Hence our claim holds true for $t+1$ $(K+1)$ as well.

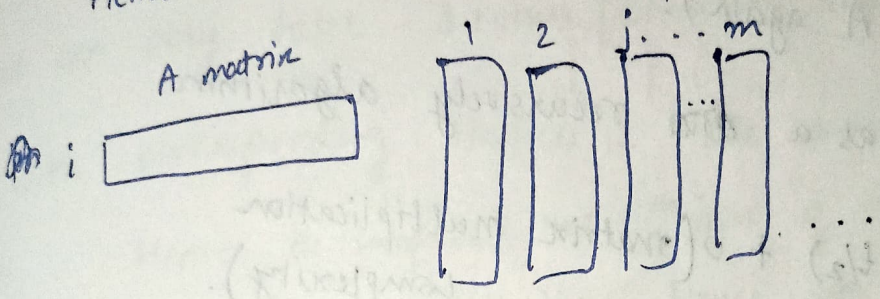
Hence by induction hypothesis we have

$$P(X_{t+1} = j \mid X_1 = i) = [A^t]_{ij}$$ where $A^t$ is $t^{th}$ power of matrix $A$.

b) We need to find the computional complexity of computing

$$P(X_{t+1} = j \mid X_1 = i) = [A^t]_{ij}$$

Hence to find $ij^{th}$ term of $A^t$. Consider 'A'.




to find $A^2$ matrix $i^{th}$ row we need to multiply the $i^{th}$ row with every column.

Computation to multiply with '1' column is 'm'.

to compute every entry of $i^{th}$ row $\Rightarrow m^2$.

Now to compute $[A^t]_{ij}$ we need $[A^{t-1}]$ $i^{th}$ row and multiply it with $j^{th}$ column of $A \Rightarrow O(m)$ - (3)

To compute $A^2$ $i^{th}$ row it took $m^2$ steps.

$A^3$ $i^{th}$ row it takes $2m^2$ steps.

Hence $A^{t-1}$ $i^{th}$ row it takes $(t-1)m^2$ steps - (4)

Total complexity $= O((t-1)m^2 + m) = O(m^2 t)$.

From (3) & (4)

(C) Another way is to compute $(A^t)$ directly and then take the $ij^{th}$ entry of the matrix.

$A^t$ can be computed by finding $A^{t/2}$ then squaring it ie.,

$$A^t = A^{t/2} \cdot A^{t/2}$$

(if t is odd we may need to multiply it with $A$ again).

Consider this as a recursively algorithm

$$T(t) = T(t/2) + O(\text{matrix multiplication complexity}).$$

Now to multiply to matrices $A \Rightarrow x \times K$, $B \Rightarrow K \times y$ it takes $O(xyK)$ becomes each entry takes $O(K)$ and we have $xy$ entries to compute. Hence $O(xyK)$.

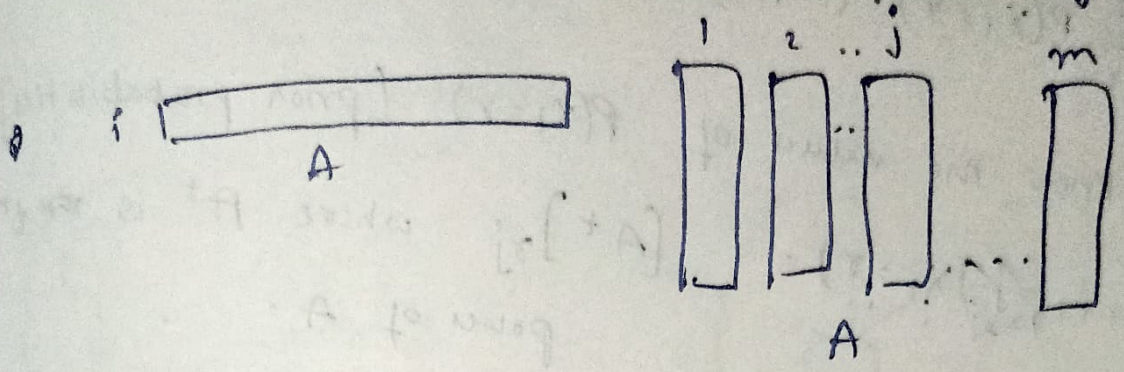
In our case $x = y = K = m$ Hence matrix multiplication complexity $= O(m^3)$

$$T(t) = T(t/2) + O(m^3)$$

$T(t) = m^3 \log_2 t$ $\Rightarrow$ then we pick $(ij)^{th}$ entry.

Hence $A^t$ computation complexity is $O(m^3 \log_2 t)$.

d) From (b) to compute $[A^2]_{ij}$ we take ith row of A and multiply it with all columns of A.




Now instead of multiply all entries of ith row (ie. m entries) we can only take s entries (which are non-zero) and multiply with corresponding indices in the column.

Hence to compute the ith row of $A^2$ we have $O(sm)$ complexity. Hence to compute.

$A^{t-1}$ ith row we have $O(sm(t-1))$.

Total complexity = $O(sm(t-1)) + O(m)$
↳ to compute $(A^t)_{ij}$ from ith row $A^{t-1}$ and $j^{th}$ column of A.

$= O(sm(t-1)) + O(m) = O(smt)$.

e) $$P(X_1 = i \mid X_{T+1} = j) = \frac{P(X_{T+1} = j \mid X_1 = i) P(X_1 = i)}{P(X_{T+1} = j)}$$

$$= \frac{P(X_{T+1} = j \mid X_1 = i) P(X_1 = i)}{\sum_{\gamma=1}^{m} P(X_{T+1} = j, X_1 = \gamma)}$$ [Marginalization].

$$= \frac{P(X_{T+1}=j \mid X_1=i)\, P(X_1=i)}{\sum_{r=1}^{m} P(X_1=r).P(X_{T+1}=j \mid X_1=r)} \quad \text{[product rule]}.$$

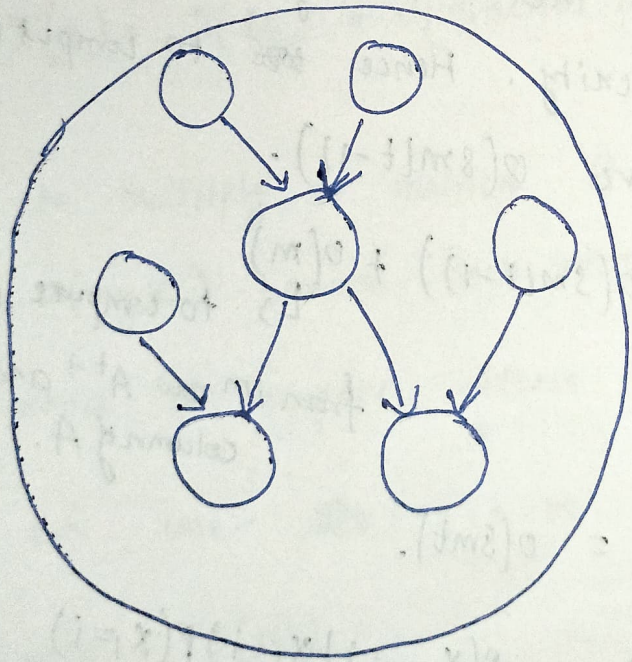
$\Rightarrow$ we know the values of $P(X_1=r)$ [prior probabilities].

$P(X_{T+1}=j \mid X_1=r) = \ldots [A^t]_{rj}$ where $A^t$ is the power of $A$.

$$= \frac{[A^t]_{ij}\, P(X_1=i)}{\sum_{r=1}^{m} P(X_1=r).\, [A^t]_{rj}}$$

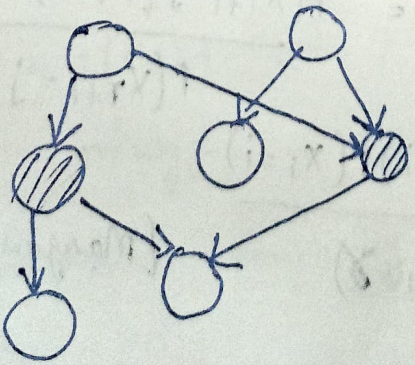
$=$

3.3)

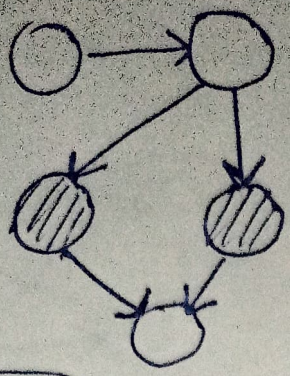


Polytree

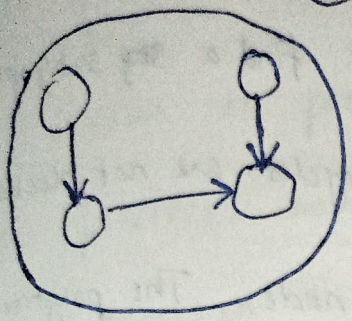
(No cycles).

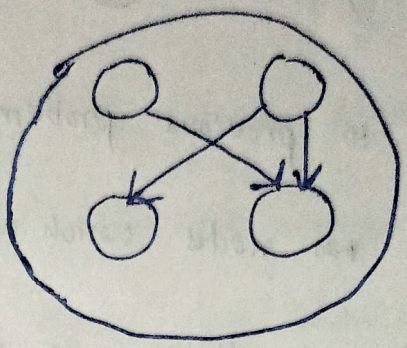


nodes which are shaded can be clustered, then it is a polytree.



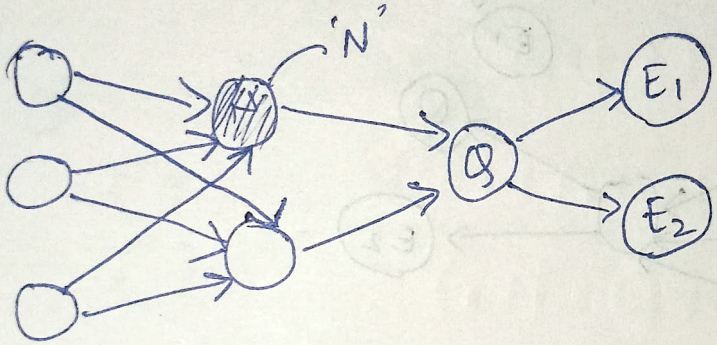
nodes which are shaded can be clustered, then it's a polytree.



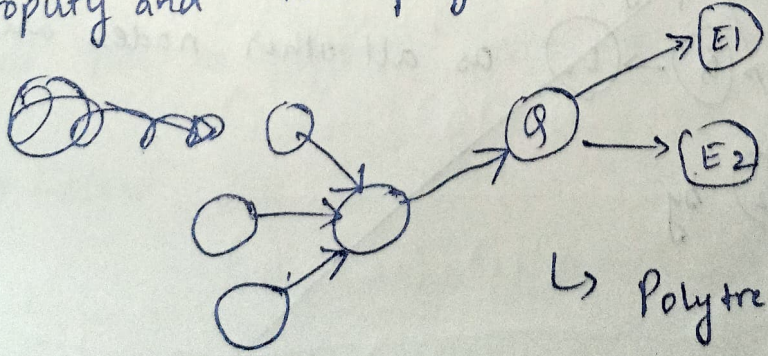


Polytree (No cycles).

3.4) a)

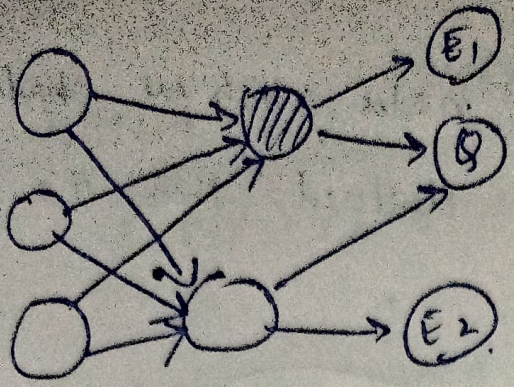



We cannot find a subgraph which is a polytree as there are nodes which are not blocked by $E_1, E_2$ using d-separation. and they are causing cycles. Hence we shade node 'N'. and then we can remove it by cutset property and run polytree

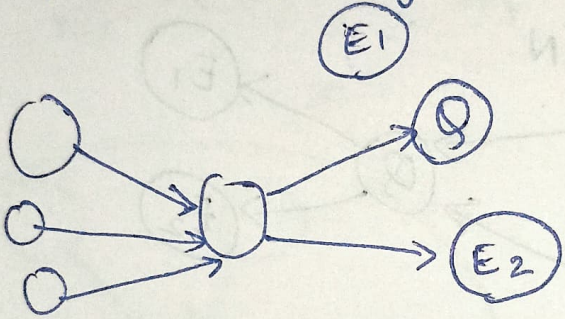


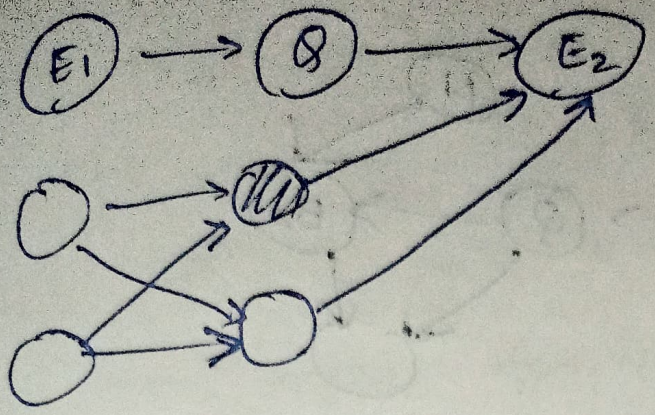

↳ Polytree.

b)

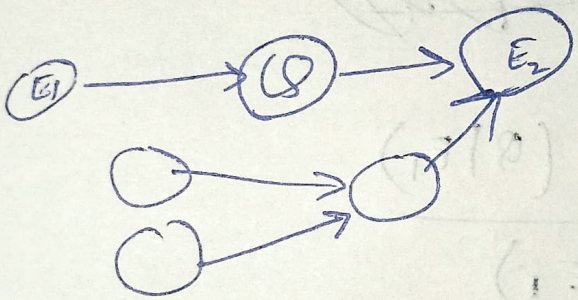


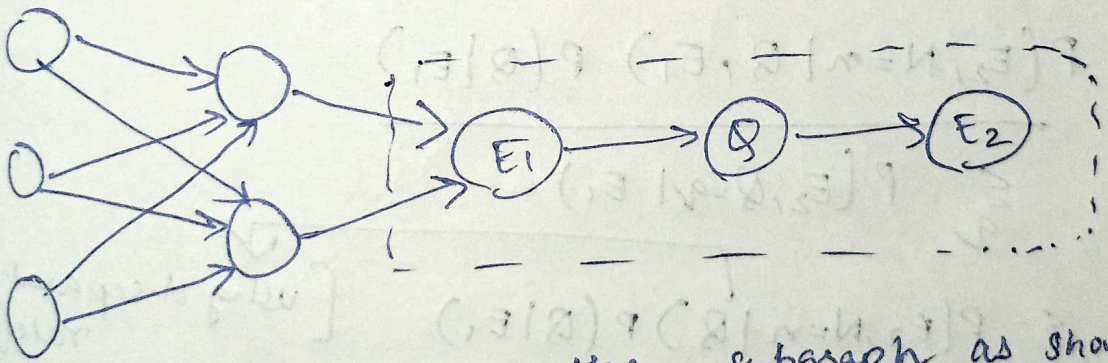

Similar to previous problem we cannot find a subgraph as polytree, as nodes which are causing cycles are not blocked by $E_1, E_2$. Hence we shade one node. The polytree obtained by cutset property is



c) 

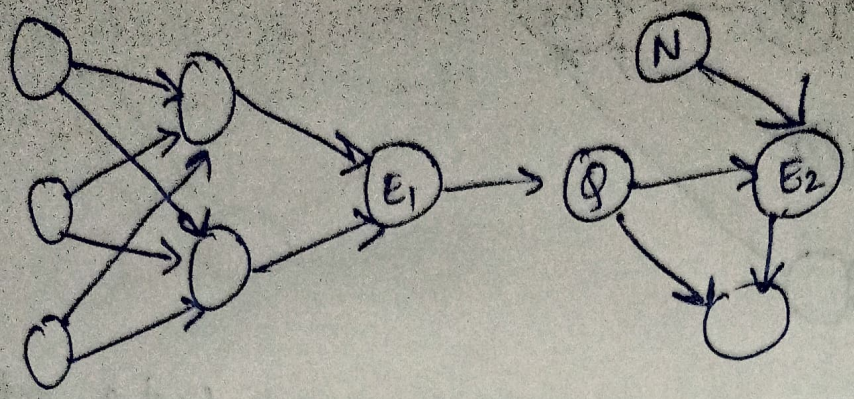

Similar to previous problem we cannot find a subgraph as polytree, as nodes which are causing cycles are not blocked by $E_1, E_2$. Hence we shade one node. The polytree obtained by mutset property.




d) 


We can consider the subgraph as shown in the dotted line as polytree, remaining part of the graph is blocked by $E_1$ node using d-separation.

c)




we need $P(Q|E_1,E_2)$, I have marked parent of $E_2$ as 'N'.

Now $P(Q|E_1,E_2)$ [Bayes rule $P(X|Y,E) = \frac{P(Y|X,E)P(X|E)}{P(Y|E)}$]

$$= \frac{P(E_2|Q,E_1)\,P(Q|E_1)}{P(E_2|E_1)}$$

$$= \frac{\sum_n P(E_2, N=n|Q,E_1)\,P(Q|E_1)}{\sum_q P(E_2, Q=q|E_1)}$$

$$= \frac{\sum_n P(E_2,N=n|Q)\,P(Q|E_1)}{\sum_q P(Q=q|E_1)\,P(E_2|E_1,Q=q)}$$ [using d-separation rule (1)]

$$= \frac{\sum_n P(E_2,N=n|Q)\,P(Q|E_1)}{\sum_q P(Q=q|E_1)\,P(E_2|Q=q)}$$ → d-separation rule (1)

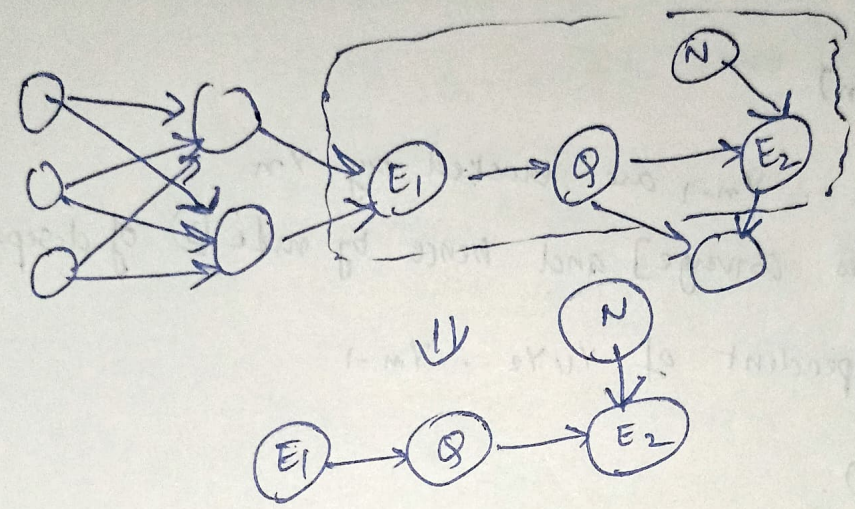
$$= \frac{\sum_n P(E_2,N=n|Q)\,P(Q|E_1)}{\sum_q \sum_n P(Q=q|E_1)\,P(E_2,N=n|Q=q)}$$

$$= \frac{\sum_n P(E_2,N=n|Q)\,P(Q|E_1)}{\sum_q \sum_n P(Q=q|E_1)\,P(N=n)\,P(E_2|Q=q,N=n)}$$

ALL VALUES CAN BE COMPUTED FROM CPT

From eq (5) we can see that

the value of $P(Q|E_1,E_2)$ only depends on node 'N', $E_1$, $E_2$ since including node 'N' does not cause any cycles we can run polytree algorithm on below dotted subgraph.




3.2)

a) $P(Y_1|X_1) = \sum_{x_0} P(Y_1, X_0 = x_0 | X_1)$ [Marginalization]

$= \sum_{x_0} P(X_0 = x_0|X_1)\, P(Y_1|X_1, X_0 = x_0)$

$= \sum_{x_0} P(X_0 = x_0)\, P(Y_1|X_1, X_0 = x_0)$ [d-separation rule (3)]

$P(X_0 = x_0)$, $P(Y_1|X_1, X_0 = x_0)$ can be computed from CPT.

b) $P(Y_1) = \sum_{x_0}\sum_{x_1} P(Y_1, X_0 = x_0, X_1 = x_1)$ [Marginalization]

$= \sum_{x_0}\sum_{x_1} P(X_0 = x_0)\, P(Y_1, X_1 = x_1|X_0 = x_0)$ [Product rule]

$= \sum_{x_0}\sum_{x_1} P(X_0 = x_0)\, P(X_1 = x_1|X_0 = x_0)\, P(Y_1|X_0 = x_0, X_1 = x_1)$ [Product rule]

$= \sum_{x_0}\sum_{x_1} P(X_0 = x_0)\, P(X_1 = x_1)\, P(Y_1|X_0 = x_0, X_1 = x_1)$ [d-separation rule (3)]

$$= \sum_{x_0} \sum_{x_1} P(X_0 = x_0) P(X_1 = x_1) P(Y_1 | X_0 = x_0, X_1 = x_1)$$

All these values can be obtained from CPT.

c) $P(X_n | Y_1, Y_2 \ldots Y_{n-1})$

$= P(X_n)$

Since $Y_1, Y_2 \ldots Y_{n-1}$ are blocked by $Y_n$ [edges converge] and hence by rule ③ of d-separation

$X_n$ is independent of $Y_1, Y_2 \ldots Y_{n-1}$

$= P(X_n)$.

$=$.

d) $P(Y_n | X_n, Y_1, Y_2 \ldots Y_{n-1})$.

$= \sum_x P(Y_n, X_{n-1} = x | X_n, Y_1, Y_2 \ldots Y_{n-1})$ [Marginalization]

$= \sum_x P(X_{n-1} = x | X_n, Y_1, \ldots Y_{n-1}) P(Y_n | Y_1 Y_2 \ldots Y_{n-1} X_{n-1} X_n)$

$\Rightarrow$ $X_{n-1}$ is conditionally independent of $X_n$ given $Y_1 Y_2 \ldots Y_{n-1}$

$=$ [d-separation rule ③, blocked by $Y_n$]

$Y_n$ is independent of $Y_1 Y_2 \ldots Y_{n-1}$ given $X_{n-1} X_n$.

$X_{n-1}$ blocks by rule ② edges converge of d-separation.

$$= \sum_{x} P(X_{n-1} = x \mid Y_1, Y_2 \ldots Y_{n-1}) \, P(Y_n \mid X_{n-1} = x, X_n)$$

↑ value given

↗ value can be obtained from CPT table.

e) $P(Y_n \mid Y_1, Y_2 \ldots Y_{n-1})$

$$= \sum_{x_n} \sum_{x} P(Y_n, X_n = x_n, X_{n-1} = x \mid Y_1, Y_2 \ldots Y_{n-1})$$

$$= \sum_{x_n} \sum_{x} P(X_{n-1} = x \mid Y_1, Y_2 \ldots Y_{n-1}) \cdot P(Y_n, X_n = x_n \mid Y_1, Y_2 \ldots Y_{n-1})$$ [product rule].

$$= \sum_{x_n} \sum_{x} P(X_{n-1} = x \mid Y_1, Y_2 \ldots Y_{n-1}) \, P(X_n = x_n \mid Y_1, Y_2 \ldots Y_{n-1}, X_{n-1}) \, P(Y_n \mid Y_1, Y_2 \ldots Y_{n-1}, X_n, X_{n-1})$$

- $X_n$ is independent of $Y_1, Y_2 \ldots Y_{n-1}$ as by d-separation rule ③ [$Y_n$ converges]

$$= \sum_{x_n} \sum_{x} P(X_{n-1} = x \mid Y_1, Y_2 \ldots Y_{n-1}) \, P(X_n = x_n) \, P(Y_n \mid Y_1, Y_2 \ldots X_n, X_{n-1})$$

$Y_n$ is independent of $Y_1, Y_2 \ldots Y_{n-1}$ given $X_{n-1}$, $X_n$

~~Hence we get~~ as $X_{n-1}$ blocks other nodes by rule ② of d-separation

$$= \sum_{x_n} \sum_{x} P(X_{n-1} = x \mid Y_1, Y_2 \ldots Y_{n-1}) \, P(X_n = x_n) \, P(Y_n \mid X_{n-1} = x_0, X_n = x_n)$$

↓ given ↓ CPT ↓ CPT

$= .$

# hw3

October 21, 2021

## 0.1 3.6 d)

### 0.1.1 Code for 3.6 b) part

- We will perform '$x$' experiments and apply the likelihood weighing formula to compute the probability for each bit.
- In each experiment we randomly choose between 0,1 values for each bit as given $P(B_i) = 0.5$ for all $i = 1$ to 10.
- We see the experiments in which B_i is actually set to 1 and multiply them with conditional probablity of Z=128 given bits, formula used is as shown:

$$P(B_i = 1|Z = 128) = \frac{\sum_{j=1}^{x} I(B_{ij}, 1) * P(Z = 128|B_{1j}, B_{2j}..B_{nj})}{\sum_{j=1}^{x} P(Z = 128|B_{1j}, B_{2j}..B_{nj})}$$

where $x$ is the number of experiments performed, $B_{ij}$ denotes the value of $B_i$ bit generated in the $j^{th}$ experiment, $I$ is the indicator function and we can estimate $P(Z/B_1, B_2, ..B_n)$ from the formula mentioned in the question.

[1]:
```
import numpy as np
import random
import matplotlib.pyplot as plt
```

[2]:
```
def compute_number_from_bits(bits):
    ans = 0
    for i in range(len(bits)):
        ans = ans*2 + bits[i]
    return ans

def get_probability_z_given_bits(z, bits, alpha):
    return ((1.0 - alpha)/(1.0 + alpha))*(alpha ** abs(z -␣
 ↪compute_number_from_bits(bits)))

def get_random_bit():
    return random.randint(0, 1)

def indicator_func(x, y):
    return (1.0 if x == y else 0.0)
```

```
def estimate_probability(bit_idx, n, z, alpha, x):
    prob_arr = []
    num = 0.0
    den = 0.0
    for j in range(int(x)):
        bits = [get_random_bit() for _ in range(int(n))]
        num = num + indicator_func(bits[n-bit_idx], 1) *␣
 ↪get_probability_z_given_bits(z, bits, alpha)
        den = den + get_probability_z_given_bits(z, bits, alpha)
        if den == 0:
            continue
        prob_arr.append(num/den)

    return prob_arr
```

[3]:
```
z = 128
x = 1e6
alpha = 0.1
n = 10
print('Z = {}, number of experiments(x) = {}, alpha = {}, number of bits (n) =␣
 ↪{}'.format(z, int(x), alpha, n))
for bit in [2, 5, 8, 10]:
    prob_arr = estimate_probability(bit, n, z, alpha, x)
    print('Value of P(B_{}=1 | Z={}) using likelihood weighting is {}'.
 ↪format(bit, z, prob_arr[-1]))
```

```
Z = 128, number of experiments(x) = 1000000, alpha = 0.1, number of bits (n) =
10
Value of P(B_2=1 | Z=128) using likelihood weighting is 0.10032771976267528
Value of P(B_5=1 | Z=128) using likelihood weighting is 0.09211866233275916
Value of P(B_8=1 | Z=128) using likelihood weighting is 0.9102160480734384
Value of P(B_10=1 | Z=128) using likelihood weighting is 0.0
```

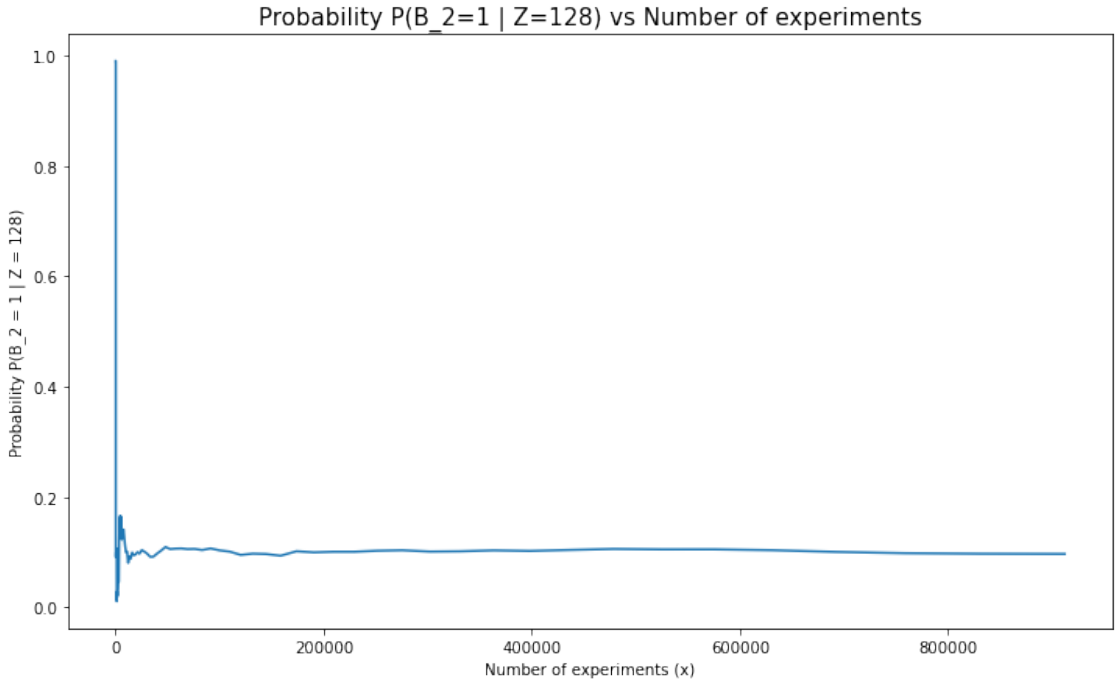
### 0.1.2 Code for 3.6 c) part

- We take the values of probability array and plot them in a log space with base 10.
- From the plot of each bit we can clearly observe that the value of probability converges to a good degree of precision.
- I am also printing the last 10 values of each probability array from which we can infer that the probability has converged to a good degree of precision.

[4]:
```
def plot_func(prob_arr, bit, num_exp):
    x_logspace = np.logspace(2, 6, 100, endpoint=False)
    x_indices = [int(idx) for idx in x_logspace]
    y_indices = [prob_arr[idx-1] for idx in x_indices]
    plt.figure(figsize=(12, 7))
```

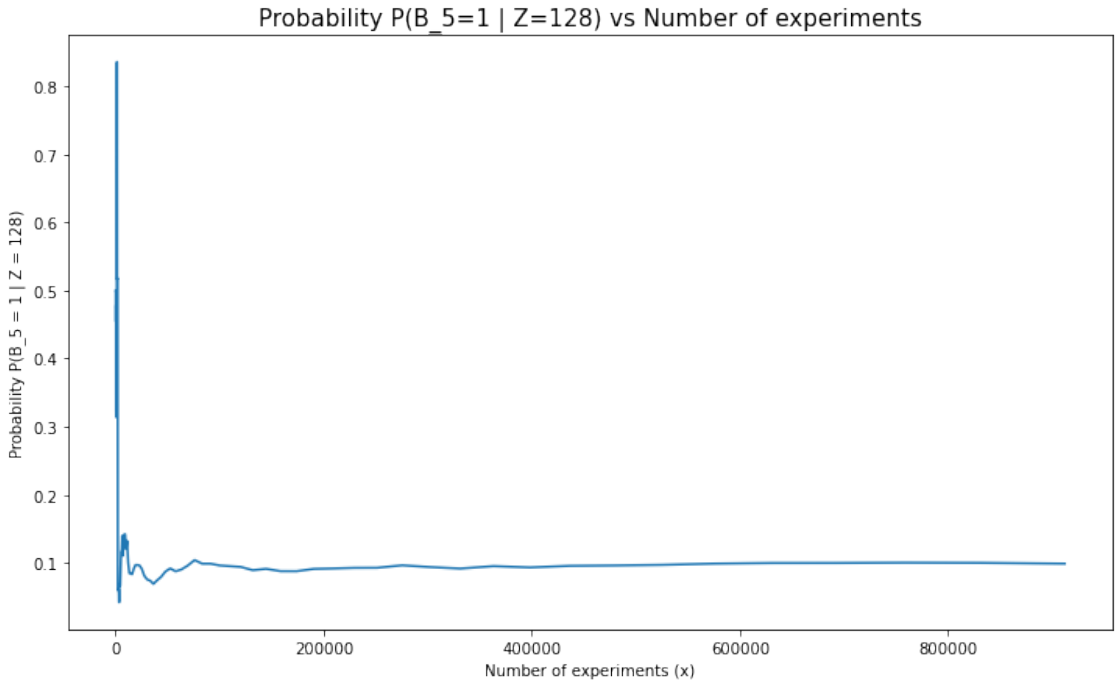
```
    plt.plot(x_indices, y_indices)
    plt.title('Probability P(B_{}=1 | Z=128) vs Number of experiments'.
 ↪format(bit, num_exp),fontsize=15)
    plt.xlabel('Number of experiments (x)')
    plt.ylabel('Probability P(B_{} = 1 | Z = 128)'.format(bit))
    plt.show()
```

[5]:
```
bit = 2
prob_arr = estimate_probability(bit, n, z, alpha, x)
plot_func(prob_arr, bit, x)
print('Out of {} experiments, the last 15 values of probability estimated for␣
 ↪bit = {} is {}'.format(x, bit, prob_arr[-15:]))
```



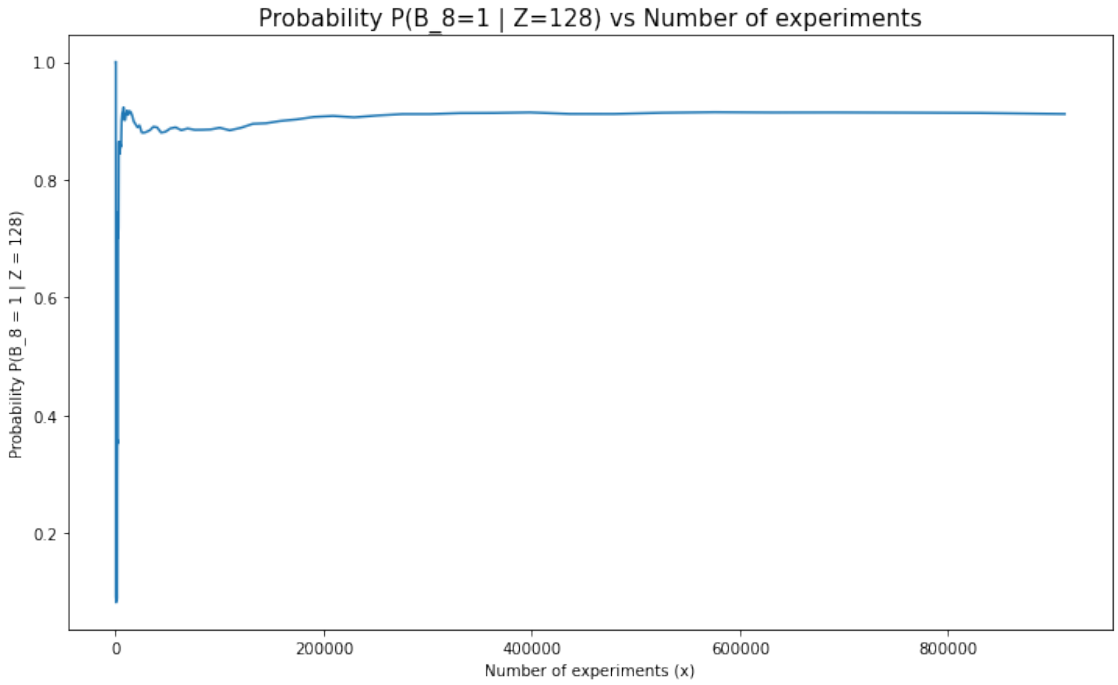

```
Out of 1000000.0 experiments, the last 15 values of probability estimated for
bit = 2 is [0.09810178306577957, 0.09801979797543843, 0.09801979797543843,
0.09801979797543843, 0.09801979797543843, 0.09801979797543835,
0.09801979797543835, 0.09801979797543835, 0.09801979797543835,
0.09801979797543835, 0.09801979797543835, 0.09801979797543835,
0.09801979797543835, 0.09801979797543835, 0.09801979797543835]
```

[6]:
```
bit = 5
prob_arr = estimate_probability(bit, n, z, alpha, x)
plot_func(prob_arr, bit, x)
print('Out of {} experiments, the last 15 values of probability estimated for␣
 ↪bit = {} is {}'.format(x, bit, prob_arr[-15:]))
```



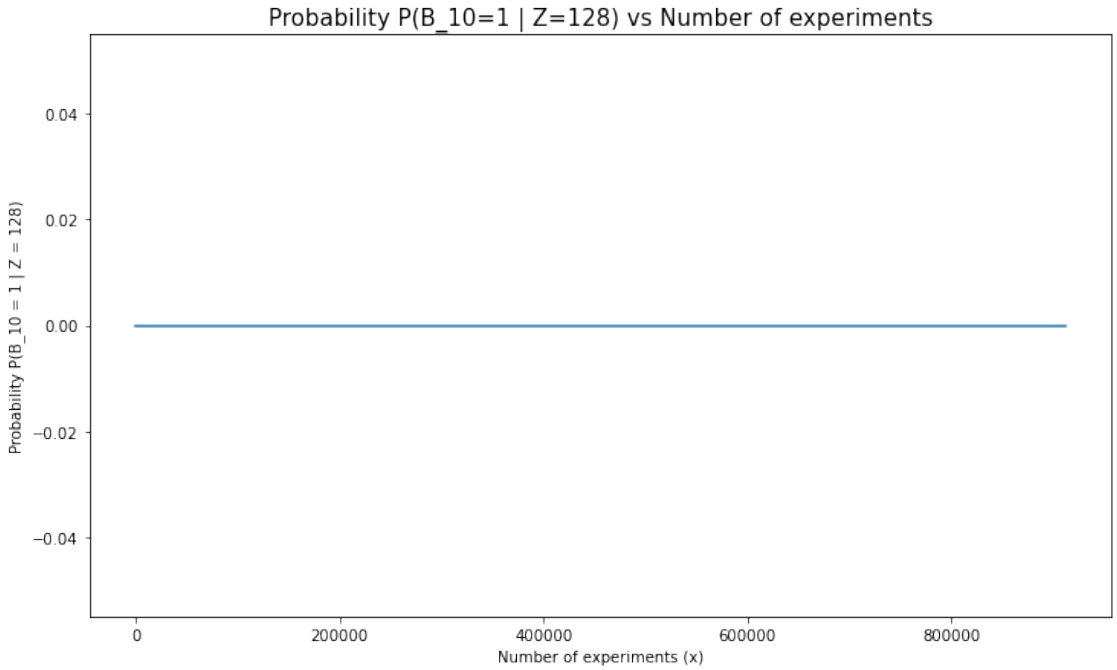

```
Out of 1000000.0 experiments, the last 15 values of probability estimated for
bit = 5 is [0.09919069007601468, 0.09919069007601468, 0.09919069007601468,
0.09919069007601468, 0.09919069007601468, 0.09919069007601468,
0.09919069007601468, 0.09919069007601468, 0.09919069007601468,
0.09919069007601468, 0.09919069007601476, 0.09919069007601476,
0.09919069007601476, 0.09919069007601476, 0.09919069007601476]
```

[7]:
```
bit = 8
prob_arr = estimate_probability(bit, n, z, alpha, x)
plot_func(prob_arr, bit, x)
print('Out of {} experiments, the last 15 values of probability estimated for␣
 ↪bit = {} is {}'.format(x, bit, prob_arr[-15:]))
```




```
Out of 1000000.0 experiments, the last 15 values of probability estimated for
bit = 8 is [0.9118333956569522, 0.9118333956569522, 0.9118333956569522,
0.9118333956569522, 0.9118333956569522, 0.9118333956569522, 0.9118333956569522,
0.9118333956569522, 0.9118333956569522, 0.9118333956569522, 0.9118333956569522,
0.9118333956569522, 0.9118333956569522, 0.9118333956569522, 0.9118333956569522]
```

[8]:
```
bit = 10
prob_arr = estimate_probability(bit, n, z, alpha, x)
plot_func(prob_arr, bit, x)
print('Out of {} experiments, the last 15 values of probability estimated for␣
 ↪bit = {} is {}'.format(x, bit, prob_arr[-15:]))
```




Out of 1000000.0 experiments, the last 15 values of probability estimated for bit = 10 is [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]