Concave Programming Upper Bounds on the Capacity of 2-D Constraints

Ido Tal Ron M. Roth

Work done while at the
Computer Science Department
Technion, Haifa 32000, Israel

2-D constraints

Example: The square constraint

- A binary $M \times N$ array satisfies the square constraint iff no two '1' symbols are adjacent on a row, column, or diagonal.
- Example:

$$\begin{array}{cccccccc} 1 & \mathbf{0} & 0 & 0 & \mathbf{0} & 1 & \mathbf{0} & 0 \\ \mathbf{0} & \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{0} & 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 1 & \mathbf{0} & 0 & 0 & 0 & 0 & 0 \end{array}$$

  - If a bold-face **0** is changed to 1, then the square constraint does not hold.

Notation for the general case

- Denote by $\mathbb{S}_M$ all the $M \times M$ arrays satisfying the constraint $\mathbb{S}$.

Capacity

Capacity Definition

- Definition:

$$\mathsf{cap}(\mathbb{S}) = \lim_{M\to\infty} \frac{1}{M^2} \cdot \log_2 |\mathbb{S}_M| \ .$$

- Intuitively: An $M \times M$ which must satisfy $\mathbb{S}$ can encode "about"

$$(\mathsf{cap}(\mathbb{S}))^{M^2}$$

  bits in it.
- Our goal: Derive an upper bound on $\mathsf{cap}(\mathbb{S})$.

Behind the scenes

(If you don't understand this slide, disregard it.)

Behind the scenes

- In the 1-D case, the capacity of a constraint is equal to the entropy of a corresponding maxentropic (and stationary) Markov chain. Namely, we calculate the entropy of a random variable, maximized over a set of probabilities.
- Essentially, we try to find a (partial) 2-D analogy.

Burton & Steif

Burton & Steif

- Theorem [Burton & Steif]: For all $M > 0$ there exists a random variable $W^{(M)}$ taking values on $\mathbb{S}_M$ such that:
  - The normalized entropy of $W^{(M)}$ approaches capacity. Namely,
$$\lim_{M\to\infty} \frac{1}{M^2} \cdot H(W^{(M)}) = \mathsf{cap}(\mathbb{S}) \ .$$
  - The probability distribution of $W^{(M)}$ is stationary.
- Notice that the theorem promises the existence of a distribution, but does not give a way to calculate it.

Bounding $H(W)$

- Recall that

$$\mathsf{cap}(\mathbb{S}) = \lim_{M\to\infty} \frac{1}{M^2} \cdot H(W^{(M)}) \ .$$

- Focus on finding an upper bound on $H(W^{(M)})$.
- Fix M and denote $W = W^{(M)}$.

Lexicographic order

Lexicographic order

Define the standard lexicographic order $\prec$ in 2-D. Namely, $(i_1, j_1) \prec (i_2, j_2)$ iff

- $i_1 < i_2$, or
- $(i_1 = i_2$ and $j_1 < j_2)$.

Example

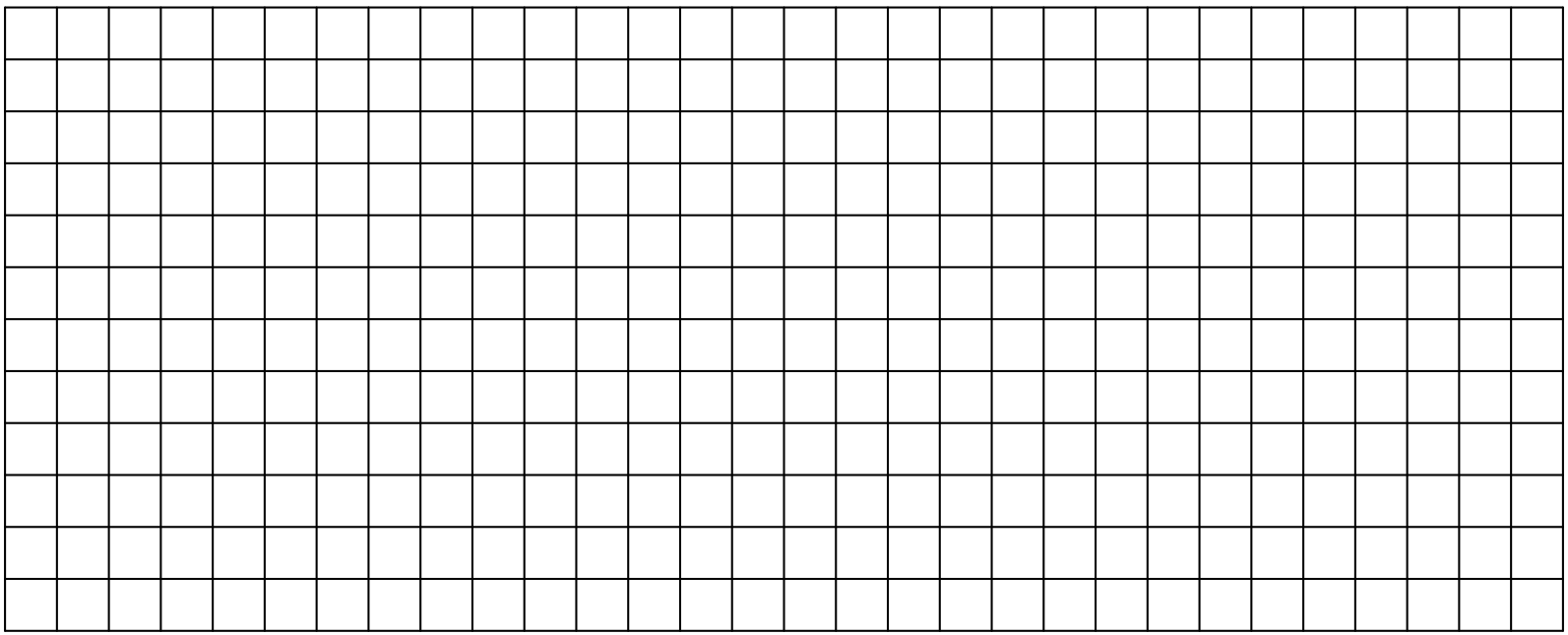
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

An entry labeled p precedes and entry labeled q iff $p < q$.

The chain rule

Define the index set $\mathsf{T}_{i,j}$ as all the indices preceding (i,j) according to $\prec$. Let B be the index set of W. By the chain rule,

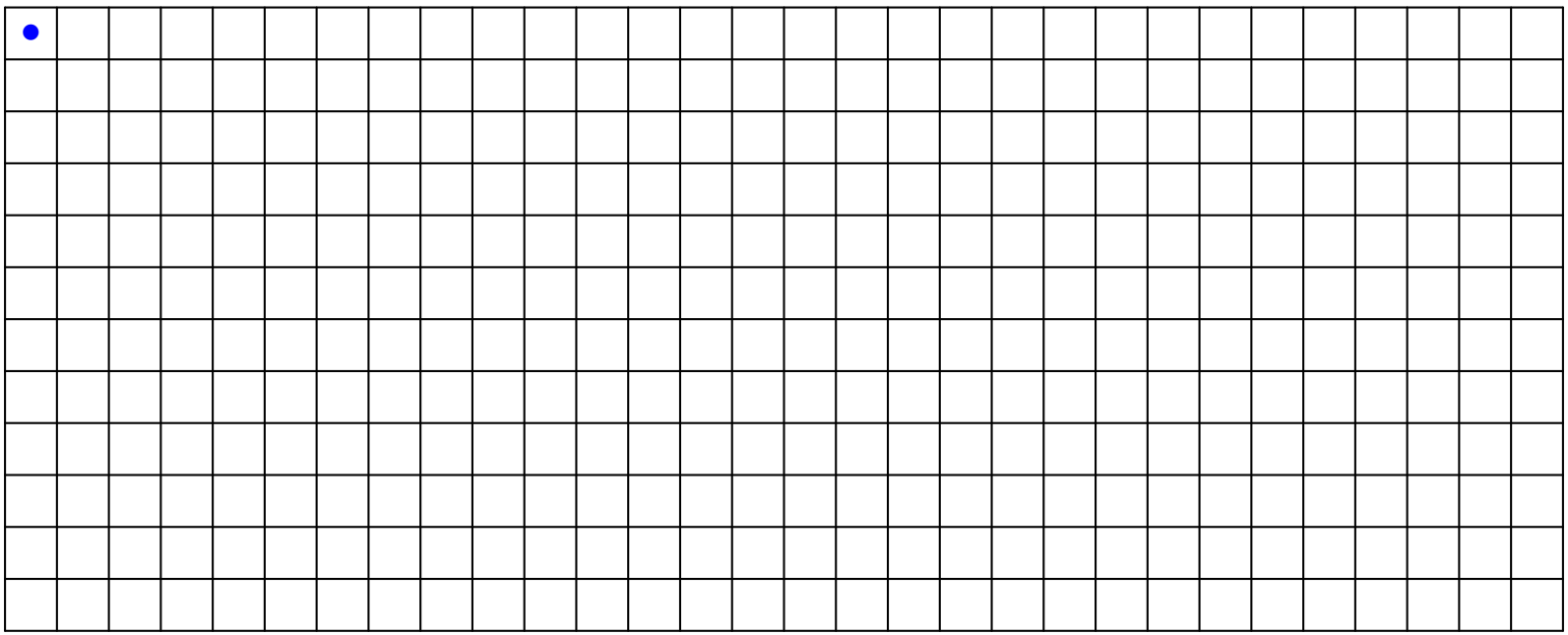
$$H(W) = \sum_{(i,j)\in\mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \ .$$



The chain rule

Define the index set $\mathsf{T}_{i,j}$ as all the indices preceding (i,j) according to $\prec$. Let B be the index set of W. By the chain rule,

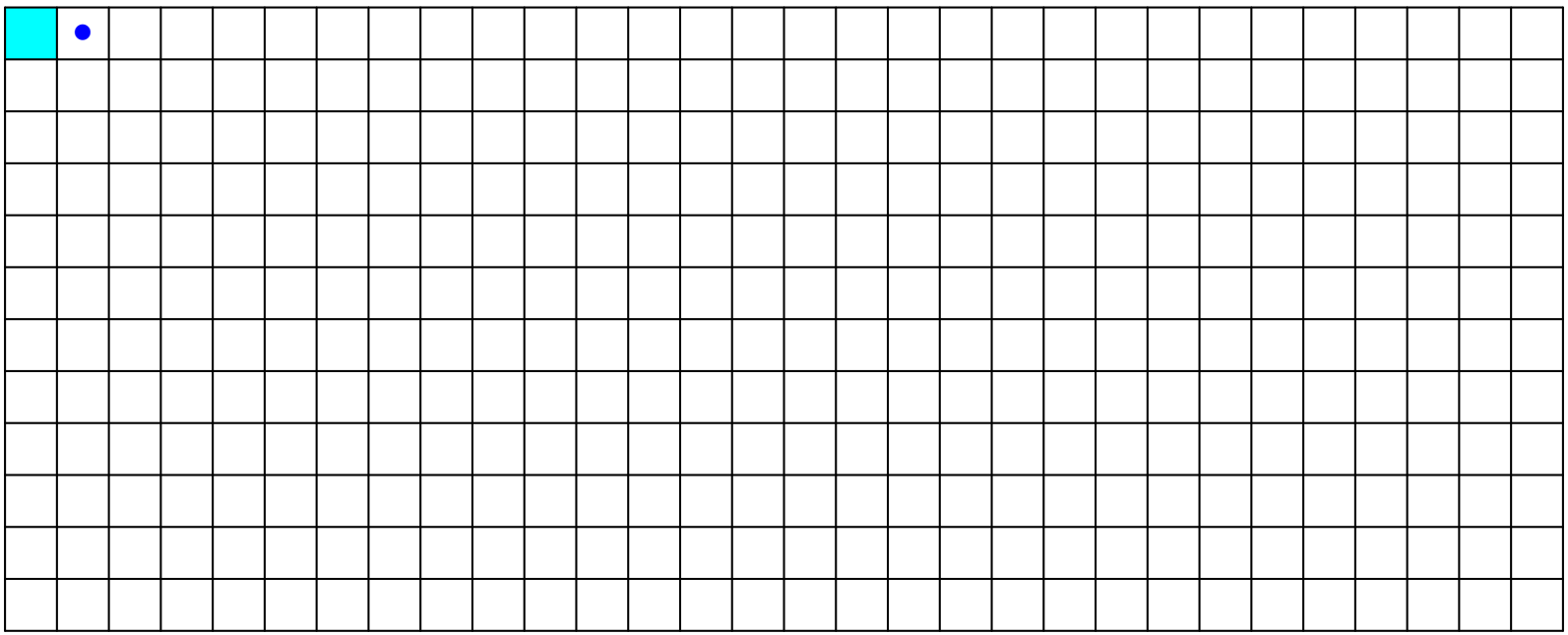
$$H(W) = \sum_{(i,j)\in\mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \ .$$



The chain rule

Define the index set $\mathsf{T}_{i,j}$ as all the indices preceding (i,j) according to $\prec$. Let B be the index set of W. By the chain rule,

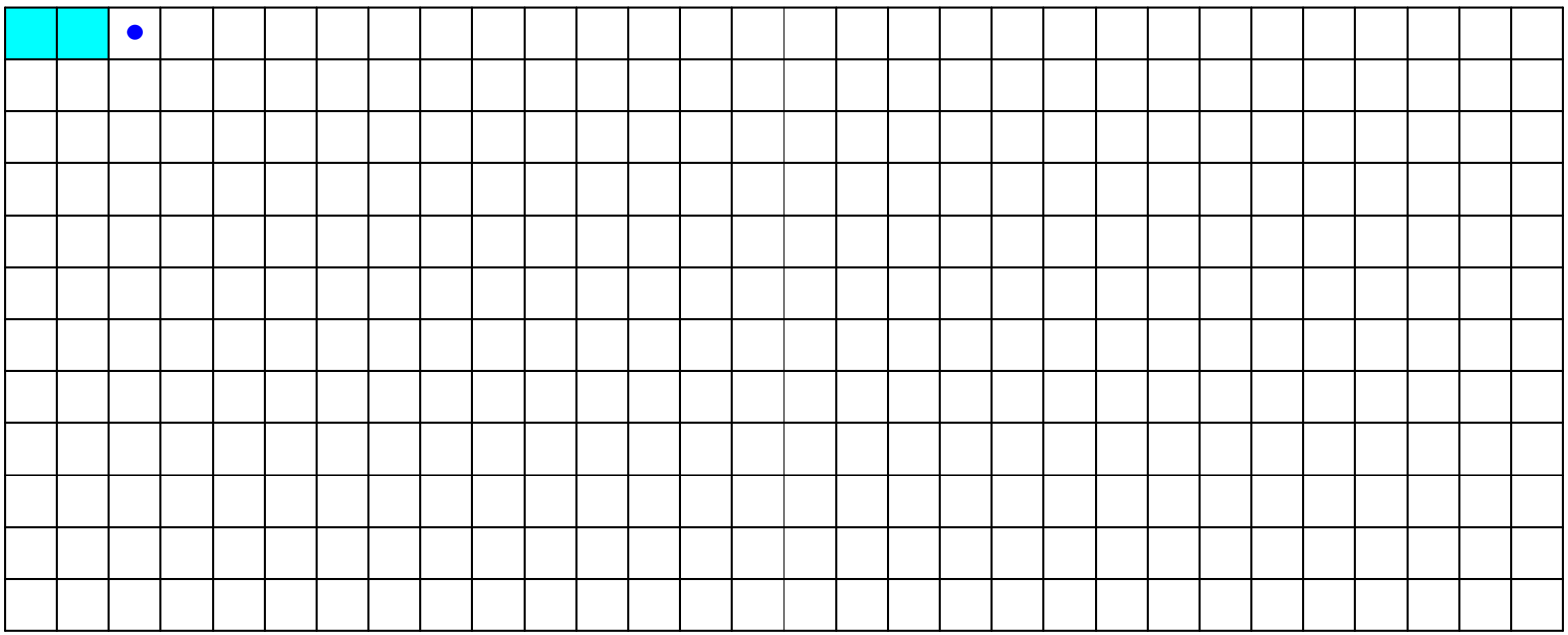
$$H(W) = \sum_{(i,j)\in\mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \ .$$



The chain rule

Define the index set $\mathsf{T}_{i,j}$ as all the indices preceding (i,j) according to $\prec$. Let B be the index set of W. By the chain rule,

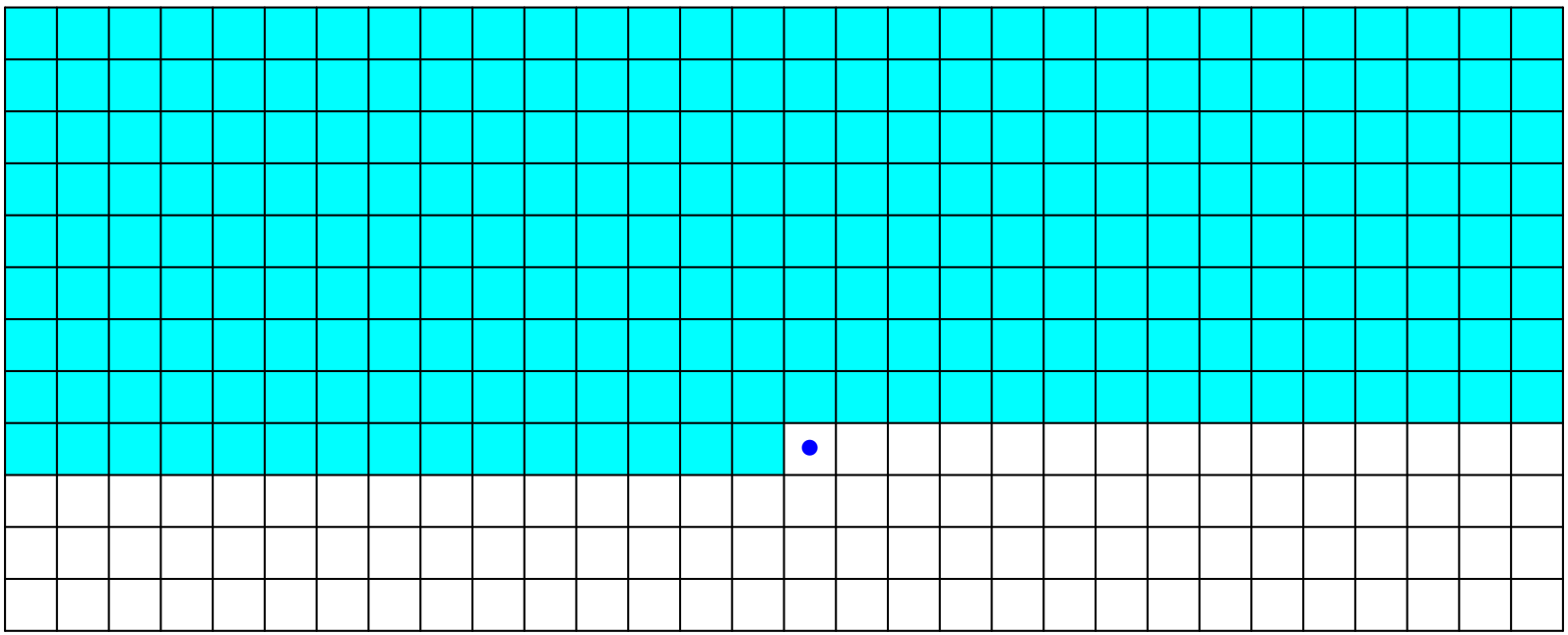
$$H(W) = \sum_{(i,j)\in\mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \ .$$



The chain rule

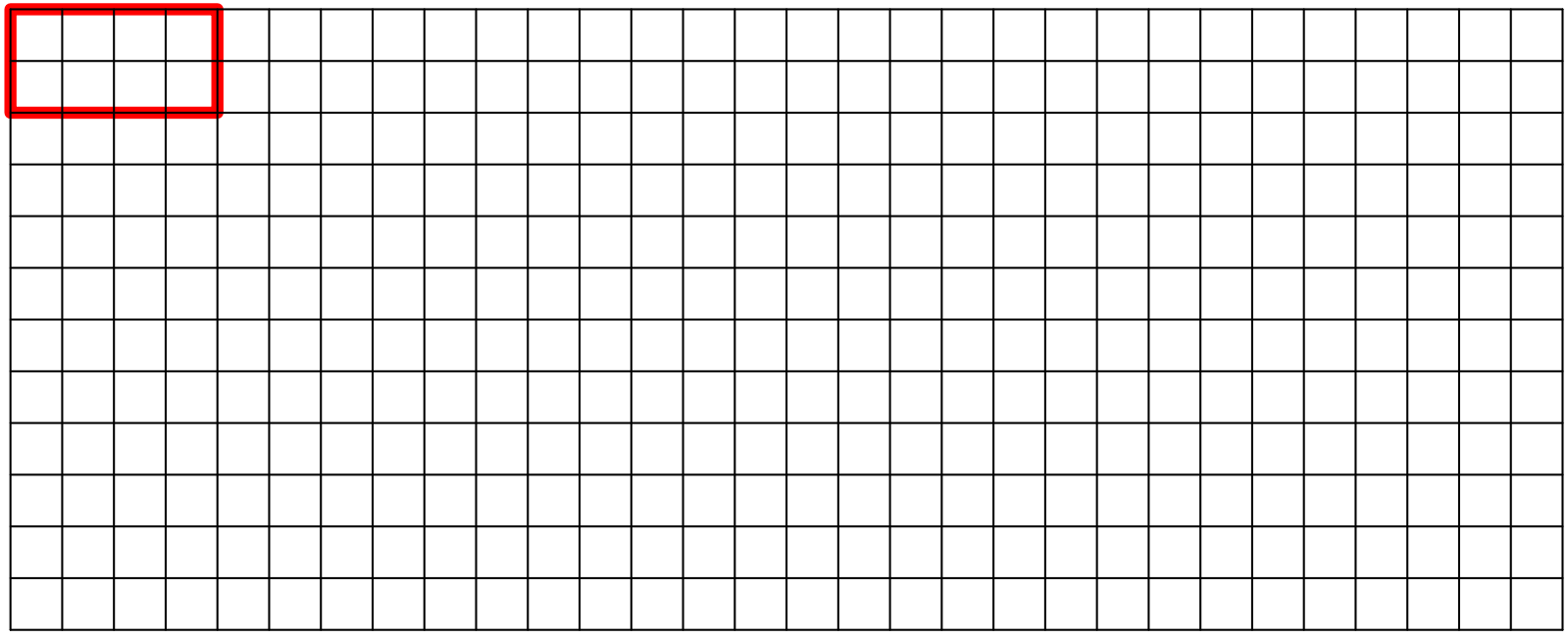
Define the index set $\mathsf{T}_{i,j}$ as all the indices preceding (i,j) according to $\prec$. Let B be the index set of W. By the chain rule,

$$H(W) = \sum_{(i,j)\in\mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \ .$$



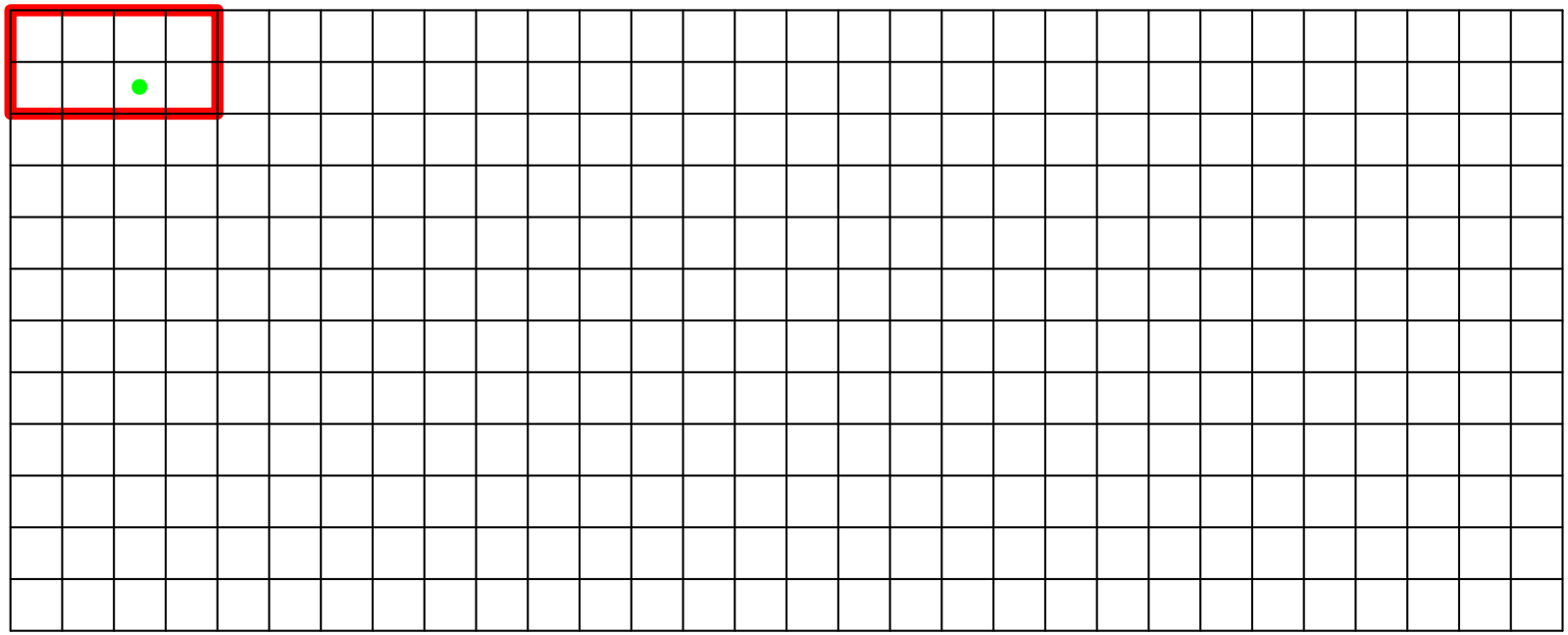
Truncating the chain

- Let Λ be a relatively small "patch", contained in B.



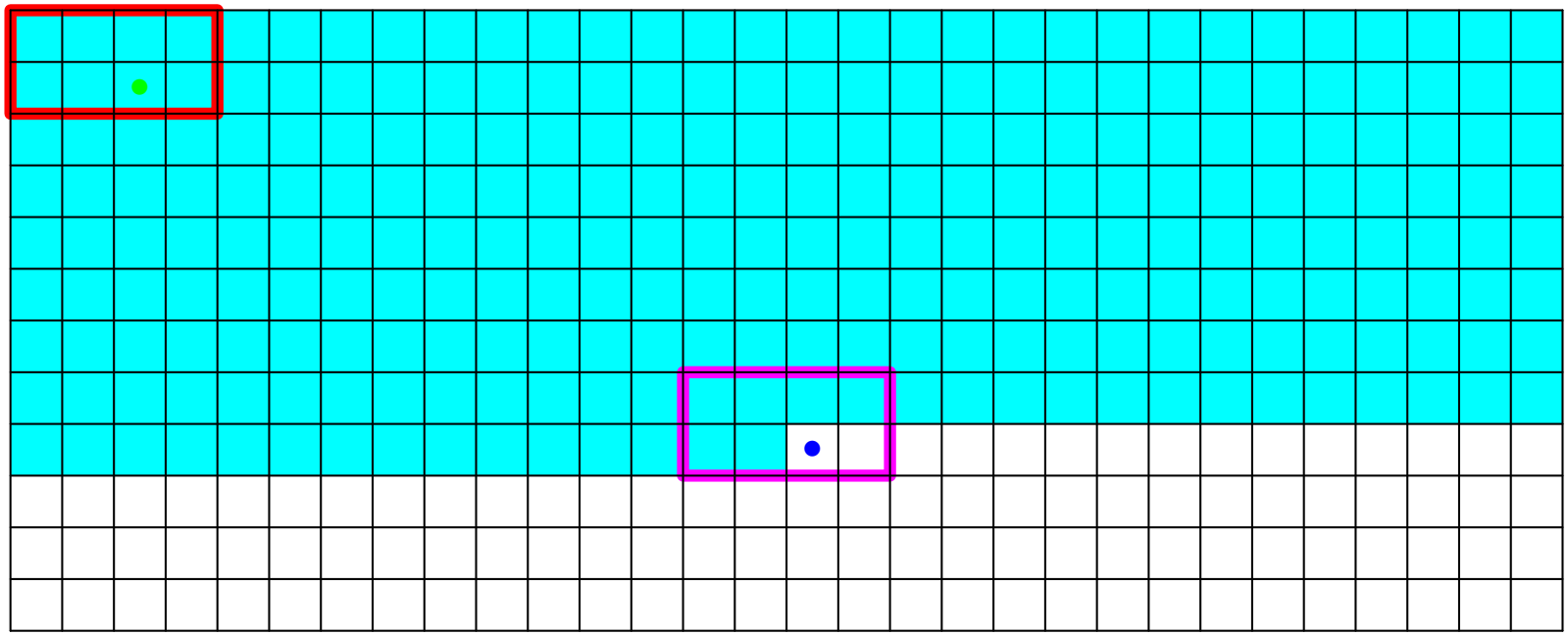
Truncating the chain

- Let Λ be a relatively small "patch", contained in B.
- Let (a, b) be an index contained in Λ.



Truncating the chain

- Let Λ be a relatively small "patch", contained in B.
- Let (a, b) be an index contained in Λ.
- Denote by $\Lambda_{i,j}$ the shifting of Λ such that (a, b) is shifted to (i, j).

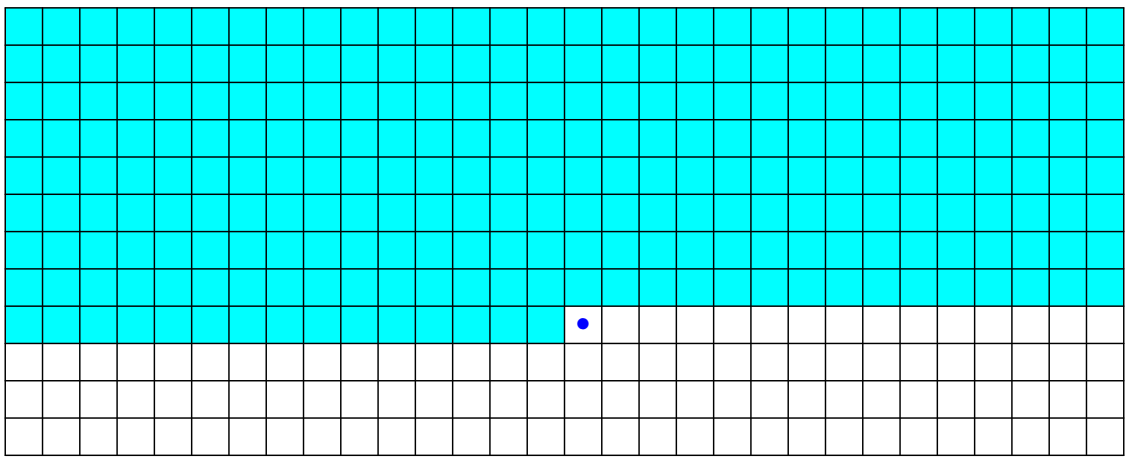


Truncating the chain

- Recall that

$$H(W) = \sum_{(i,j)\in \mathsf{B}} H(W_{i,j} | W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \, .$$

- Previously: Condition on all of the preceding entries, $W[\mathsf{T}_{i,j} \cap \mathsf{B}]$.
- Now: Condition only on preceding entries contained in the patch, $W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}]$.

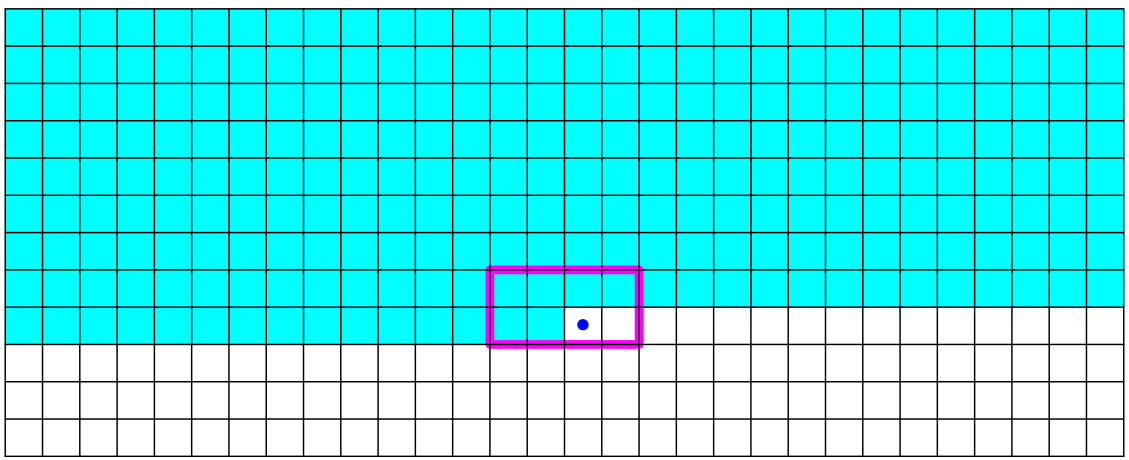


Truncating the chain

- Recall that

$$H(W) = \sum_{(i,j)\in \mathsf{B}} H(W_{i,j} | W[\mathsf{T}_{i,j} \cap \mathsf{B}]) \, .$$

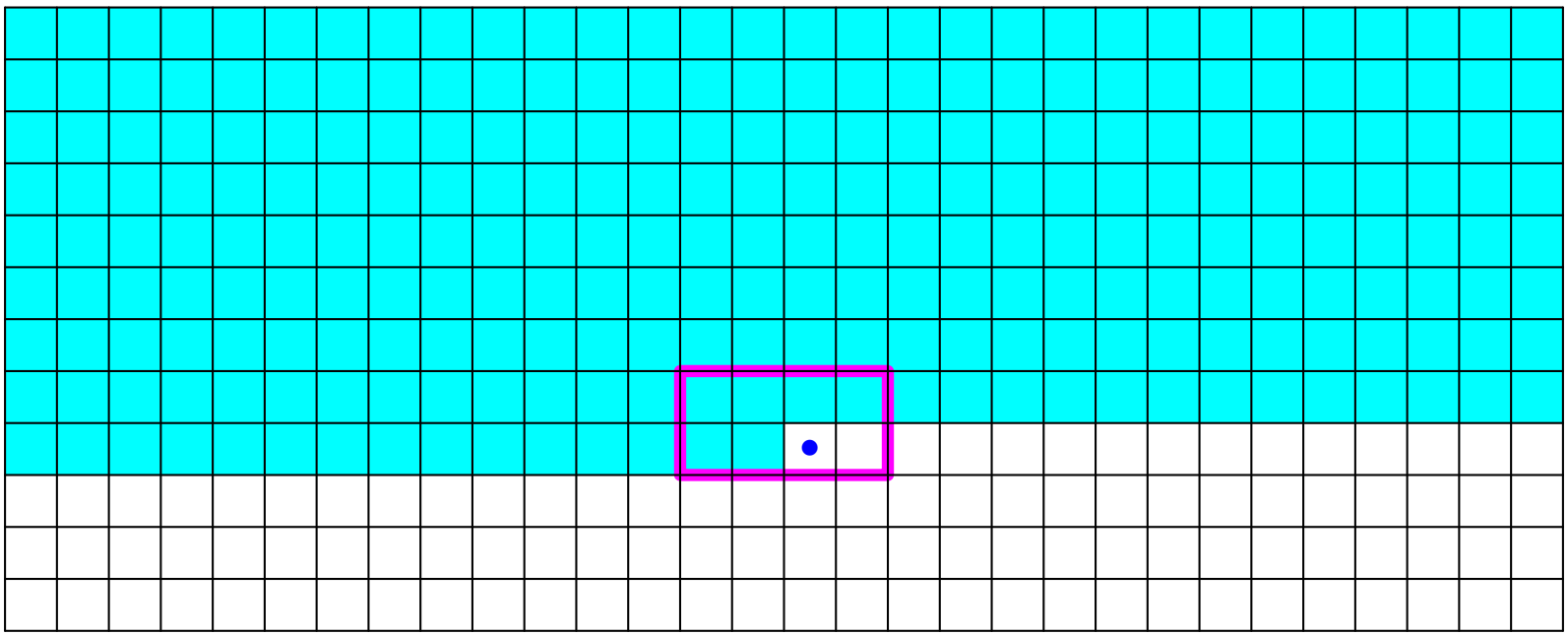
- Previously: Condition on all of the preceding entries, $W[\mathsf{T}_{i,j} \cap \mathsf{B}]$.
- Now: Condition only on preceding entries contained in the patch, $W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}]$.



Truncating the chain, illustrated

Conditioning on only a subset of the preceding entries gives us an upper bound.

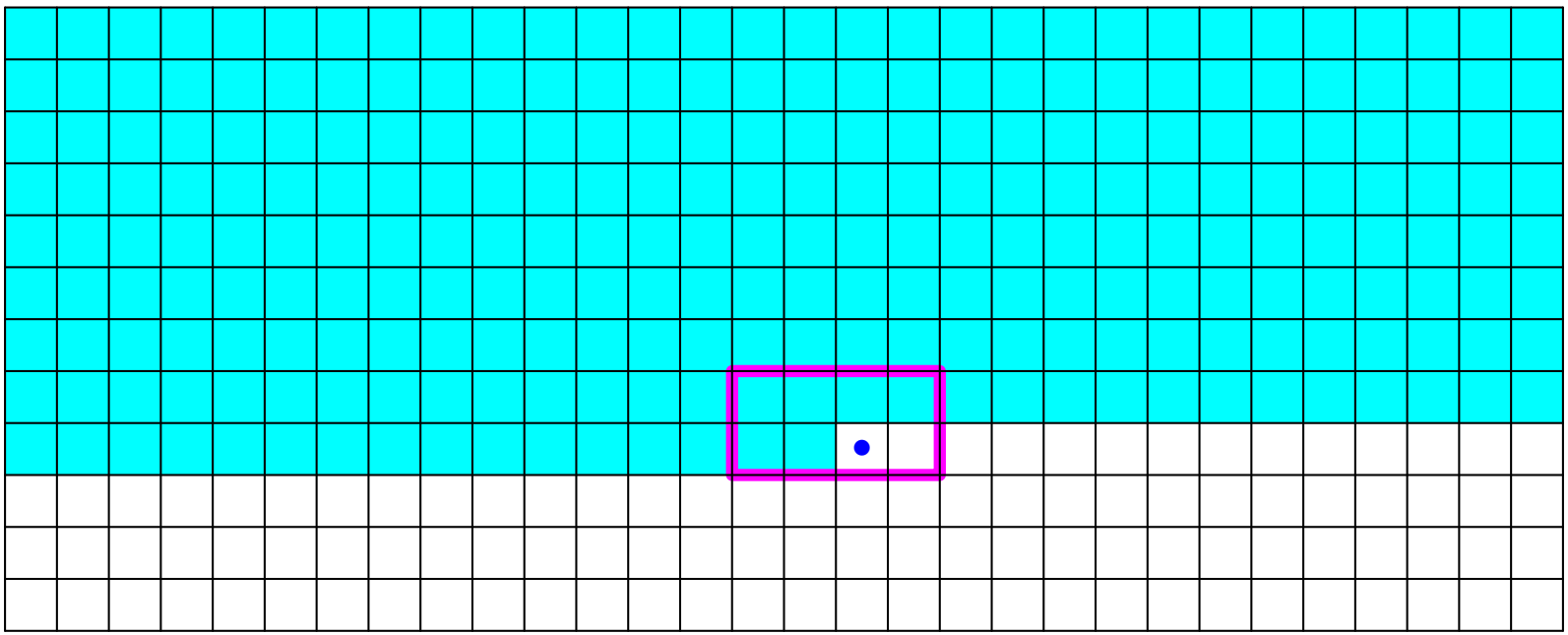
$$H(W) \leq \sum_{(i,j)\in \mathsf{B}} H(W_{i,j} | W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}]) \ .$$



Truncating the chain, illustrated

Conditioning on only a subset of the preceding entries gives us an upper bound.

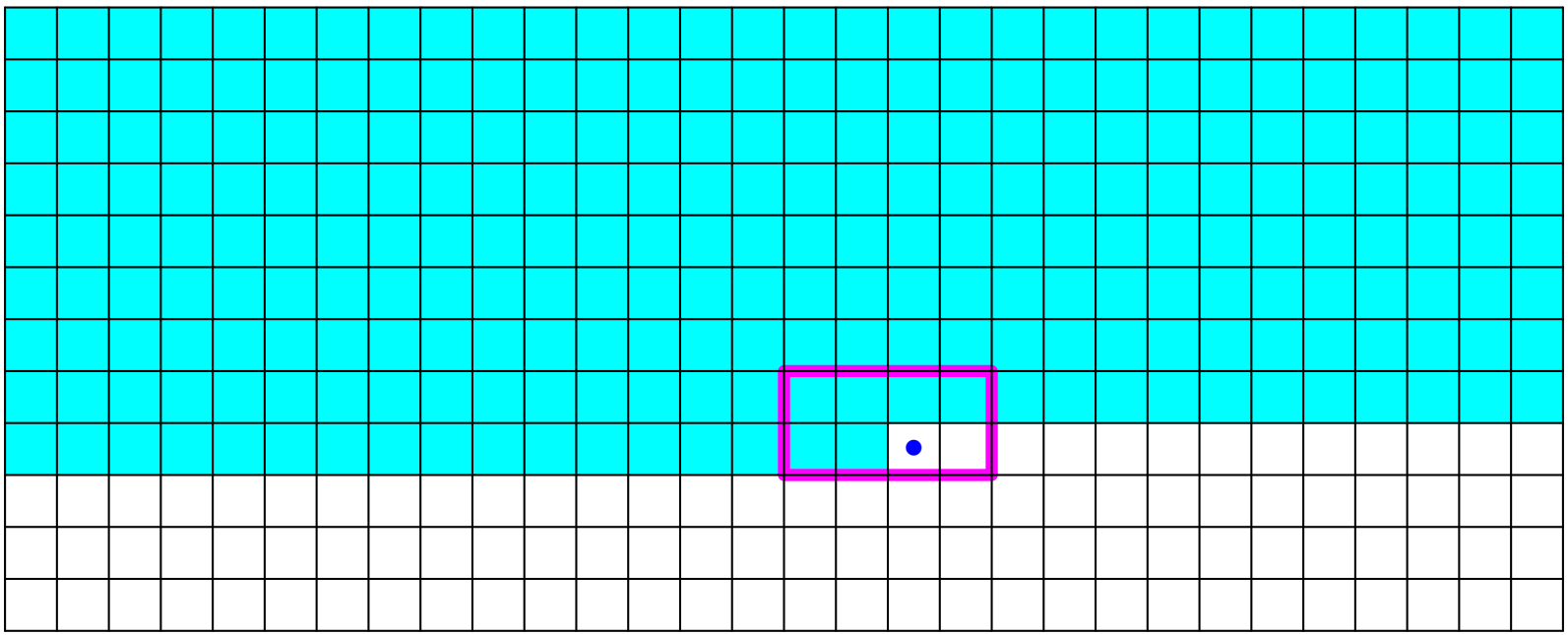
$$H(W) \leq \sum_{(i,j)\in \mathsf{B}} H(W_{i,j} | W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}]) \ .$$



Truncating the chain, illustrated

Conditioning on only a subset of the preceding entries gives us an upper bound.

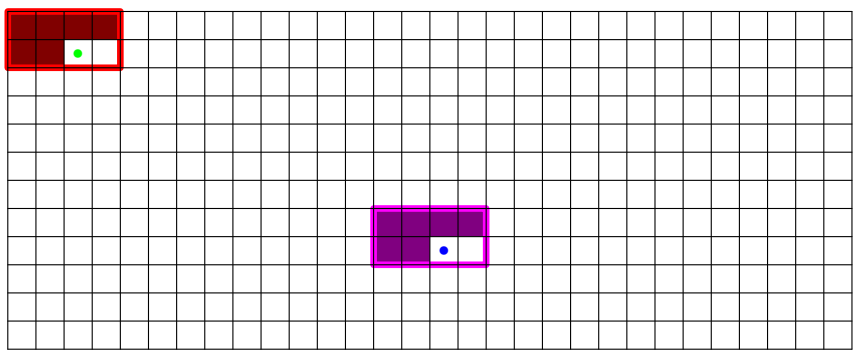
$$H(W) \leq \sum_{(i,j)\in \mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}]) \ .$$



Truncating the chain rule

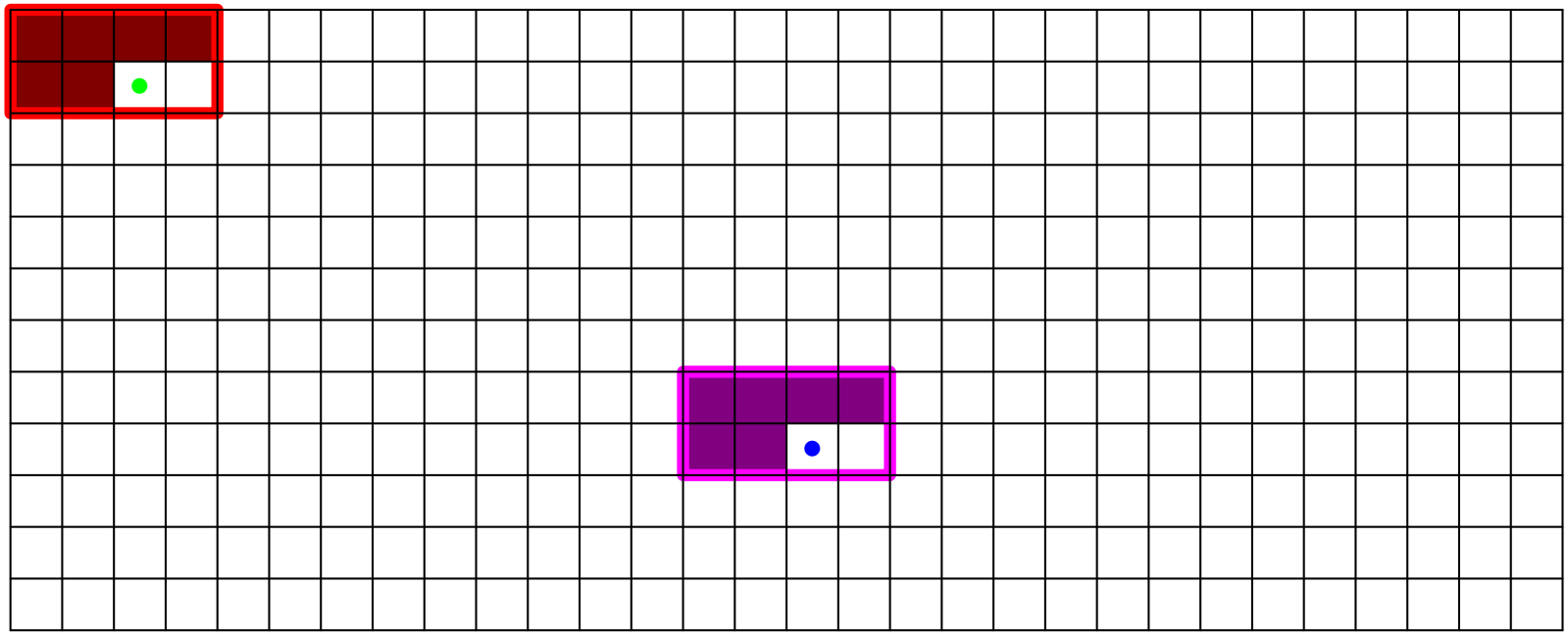
- Recall that W is stationary.
- Thus, for all (i,j) such that the patch $\Lambda_{i,j}$ is contained inside the array,

$$H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}]) = H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda]) \ .$$



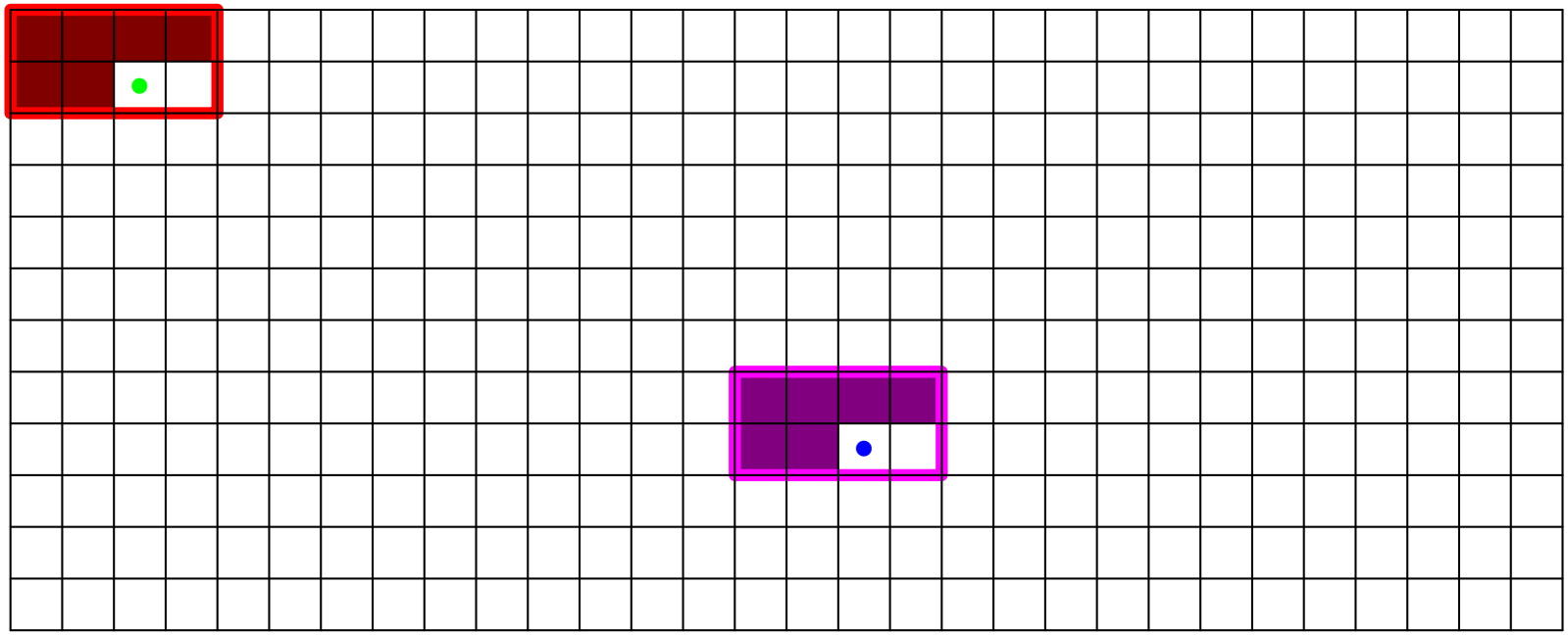
$$H(W) \leq \sum_{(i,j)\in \mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}])$$
$$\approx M^2 \cdot H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda]) \ .$$

As long as we're not near the border,
the same term is summed over and over.



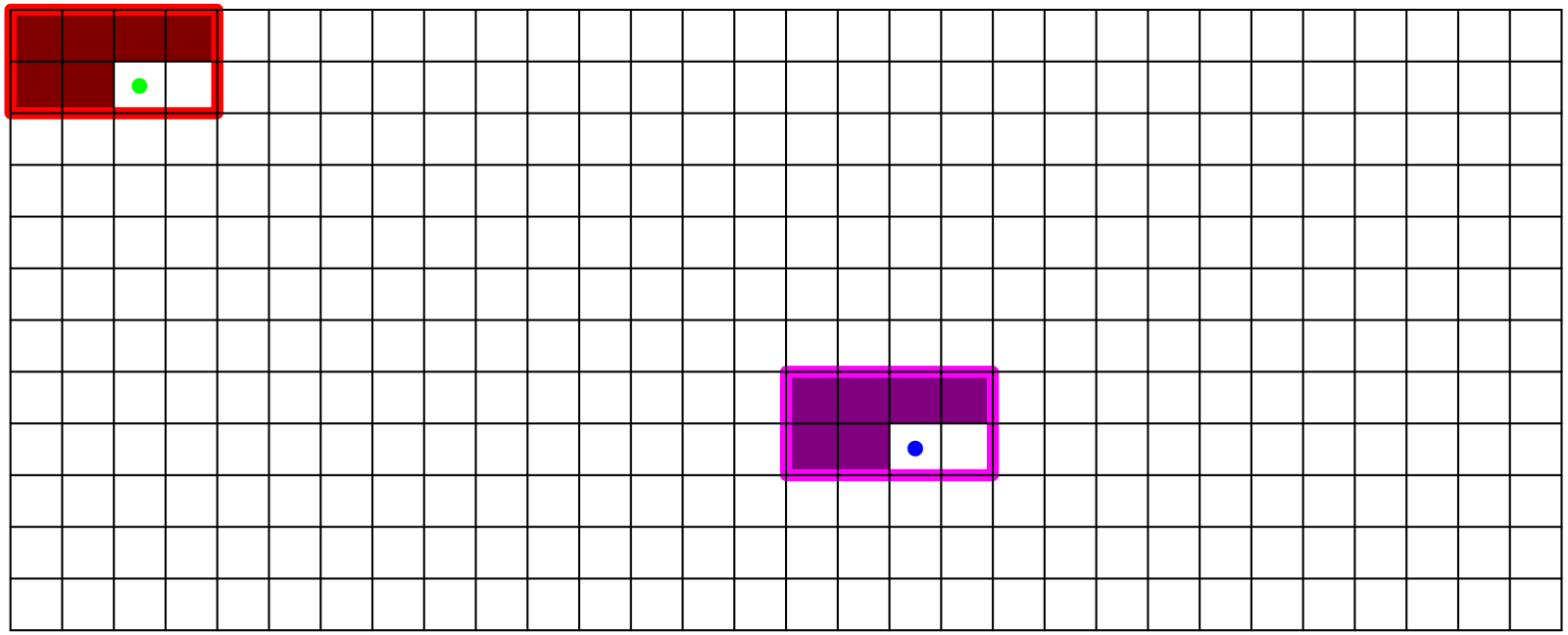
$$H(W) \leq \sum_{(i,j)\in \mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}])$$
$$\approx M^2 \cdot H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda]) \ .$$

As long as we're not near the border,
the same term is summed over and over.



$$H(W) \leq \sum_{(i,j)\in \mathsf{B}} H(W_{i,j}|W[\mathsf{T}_{i,j} \cap \mathsf{B} \cap \Lambda_{i,j}])$$
$$\approx M^2 \cdot H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda]) \ .$$

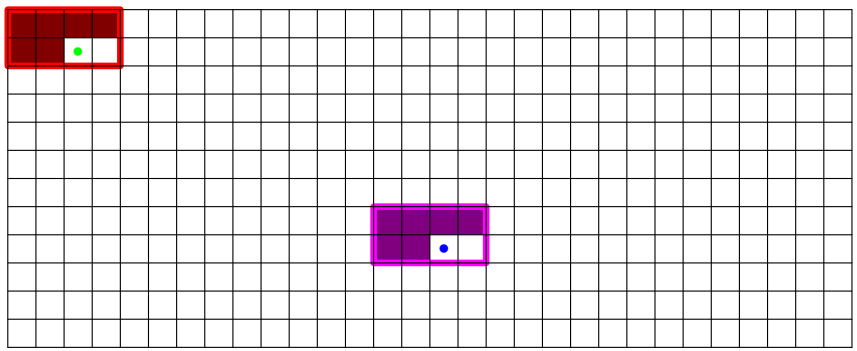
As long as we're not near the border,
the same term is summed over and over.



Truncating the chain rule

- Thus, a simple derivation gives

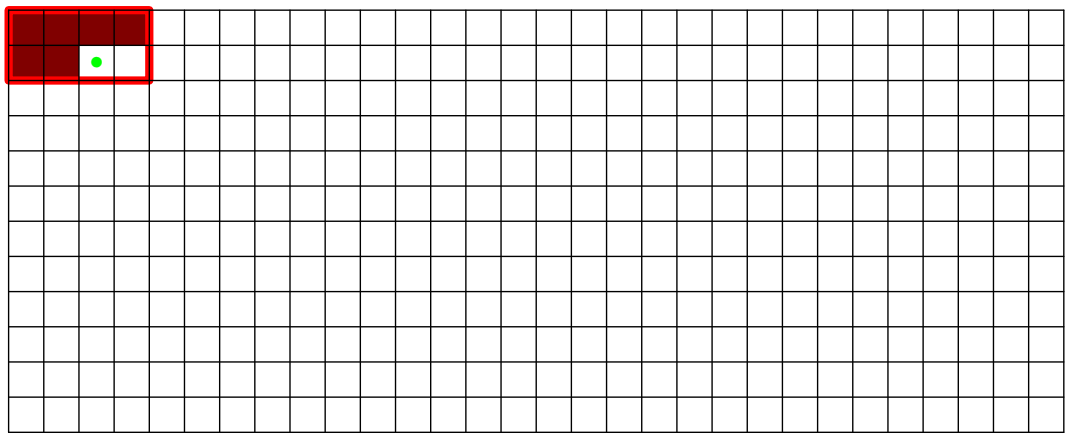
$$\frac{H(W)}{M^2} \leq H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda]) + O(1/M) \ .$$



Unknown probability distribution

$$\underbrace{H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda])}_{\clubsuit} \ .$$

- In order to calculate $\clubsuit$, we must know the probability distribution of $W[\Lambda]$.
- We don't know the probability distribution of $W[\Lambda]$, but we do know some of its properties.



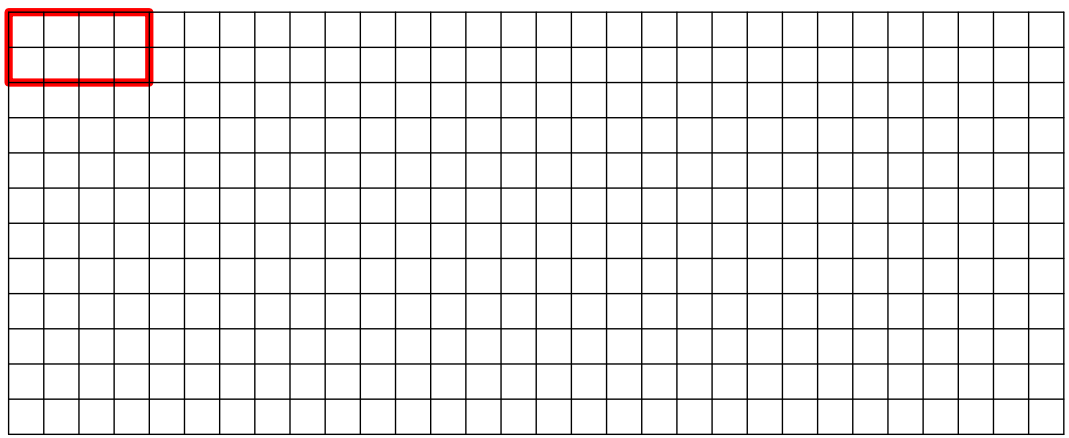
Known properties of the probability distribution (1)

Trivial knowledge

Let x be a realization of $W[\Lambda]$, with positive probability p_x.

- We know that x satisfies the constraint $\mathbb{S}$.
- We know that

$$\sum_x p_x = 1 \, .$$



Known properties of the probability distribution (2)

Vertical stationarity

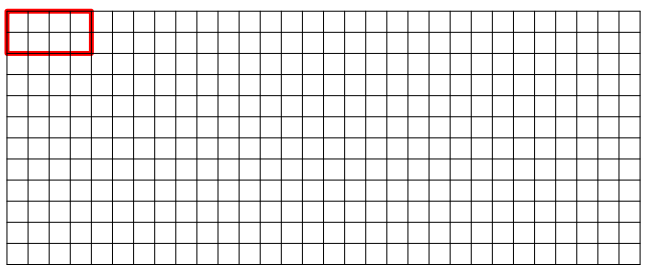
- Since W is stationary, $W[\Lambda]$ is stationary as well.
- Thus, for example,

$$P\left(W[\Lambda] = \begin{array}{|c|c|c|c|}\hline 1&0&0&1\\\hline *&*&*&*\\\hline\end{array}\right) = P\left(W[\Lambda] = \begin{array}{|c|c|c|c|}\hline *&*&*&*\\\hline 1&0&0&1\\\hline\end{array}\right) .$$

- The above can be written as

$$\sum_{x\in A} p_x = \sum_{x\in B} p_x ,$$

where x is in A (B) iff its first (second) row is 1 0 0 1.



Known properties of the probability distribution (3)

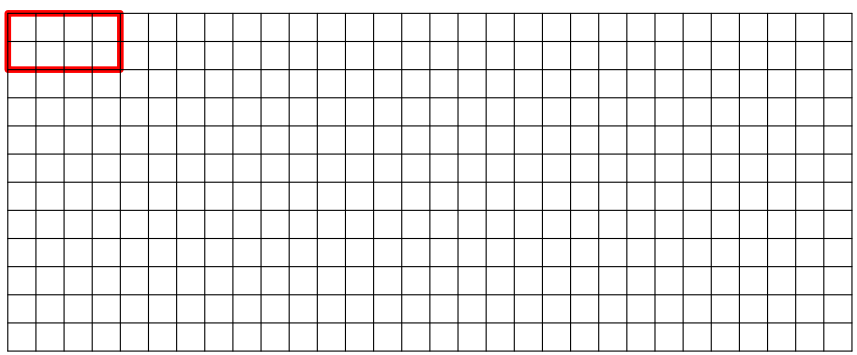
Horizontal stationarity

- Another example

$$P\left(W[\Lambda] = \begin{array}{|c|c|c|c|}\hline 1&0&0&*\\\hline 0&0&0&*\\\hline\end{array}\right) = P\left(W[\Lambda] = \begin{array}{|c|c|c|c|}\hline *&1&0&0\\\hline *&0&0&0\\\hline\end{array}\right) .$$

- Again, both sides are marginalizations of $(p_x)_x$).

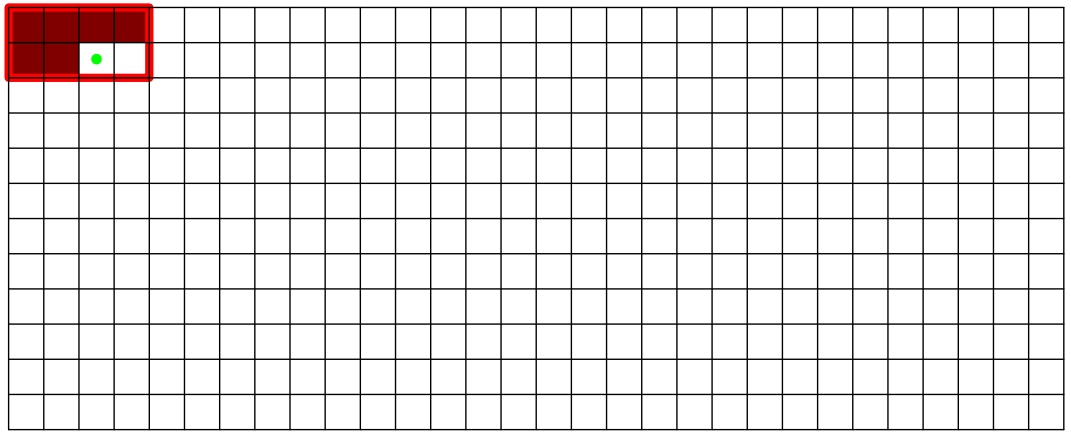
To sum up, the probabilities $(p_x)_x$ satisfy a collection of linear equalities and inequalities.



An upper bound

$$\underbrace{H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda])}_{\clubsuit} \ .$$

- We don't know the probability distribution of $W[\Lambda]$, but we do know some of its properties.
- So, let us choose the probability distribution that maximizes $\clubsuit$ and is subject to these properties. This is an instance of convex programming.



Conclusion

$$\frac{H(W)}{M^2} \leq \underbrace{H(W_{a,b}|W[\mathsf{T}_{a,b} \cap \mathsf{B} \cap \Lambda])}_{\clubsuit} + O(1/M) \ .$$

- Using convex programming, we can find an upper bound on $\clubsuit$.
- Since $\mathsf{cap}(\mathbb{S}) = \lim_{M\to\infty} \frac{H(W)}{M^2}$, this upper bound leads to an upper bound on $\mathsf{cap}(\mathbb{S})$.
- Improvements to the basic bound:
  - Combine between different choices of (a, b).
  - Combine between different choices of a precedence relation.
  - Use inherent symmetries of the constraint.

More than two dimensions

Notice that all of the above can be generalized to 3-D, 4-D, ... constraints.

Numerical results

Constraint	r	s	Upper bound	Comparison	Lower bound
$(2,\infty)$-RLL	3	8	0.4457	0.4459	0.44420
$(3,\infty)$-RLL	4	8	0.36821	0.3686	0.36562
$(0,2)$-RLL	3	5	0.816731	0.817053	0.81600
n.i.b.	3	4	0.92472	0.927855	0.92264

- The patch Λ is an $r \times s$ array.
- Larger values of r and s produce better bounds, but result in a larger optimization problem (which takes more time to solve).