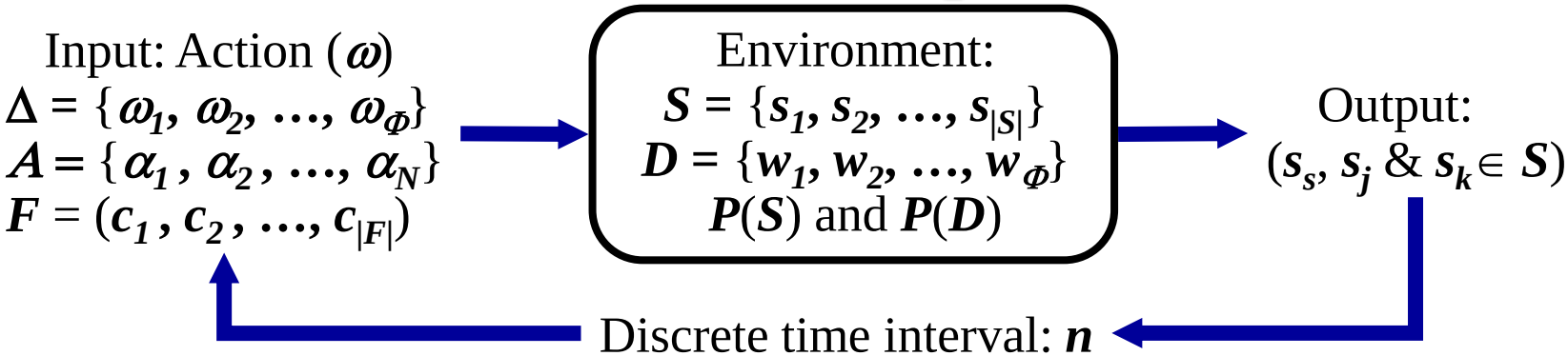
S&M
Split and Merge Compression Algorithm

By

Abdullah Hashim

S&M algorithm: The Finite Case

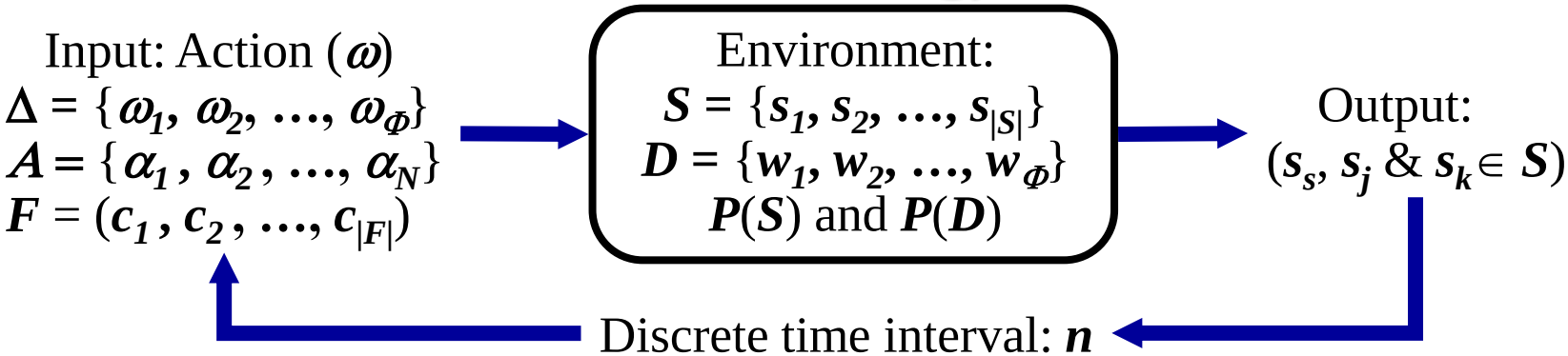
S&M algorithm: the finite case
-The Concept-



Satellite Sets ({}): In the asymptotic case, it was assumed that, the size of the sub sets $|s_i| >> |F|$, (i =1, …, $|S|$), therefore, the process of splitting can continued with no limitation. However, practically $|s_i|$ have a finite size, usually < $|F|$. The splitting process therefore is halted when $|s_i| = 1$. To overcome this problem, the concept of satellite set is introduced. A satellite is a data-empty set, carries no information, (a ***null*** set denoted by ***{}***) corresponding to an empty word, (a ***null*** word denoted by Λ), in the dictionary D. A set with one word ($|s_i| = 1$) known as a ***singleton*** set. A singleton set with satellites called parent singleton of a group of satellite. The number of satellites in the group called ***block length*** of s_i and denoted by $SBL(s_i)$ is equal to (2^r-1), (where r is a positive integer). $\{SBL(s_i) + 1\}$ is called the family block length of s_i and denoted by $\{FBL(s_i)\}$,

where: $1 \leq FBL(s_i) \leq (|S|/2)$; $FBL(s_i) = 2^r$, $r = 0, 1, 2, \ldots, \log_2(|S|/2)$.

S&M algorithm: the finite case
-The Strategy-



Matching, Merging and Splitting:

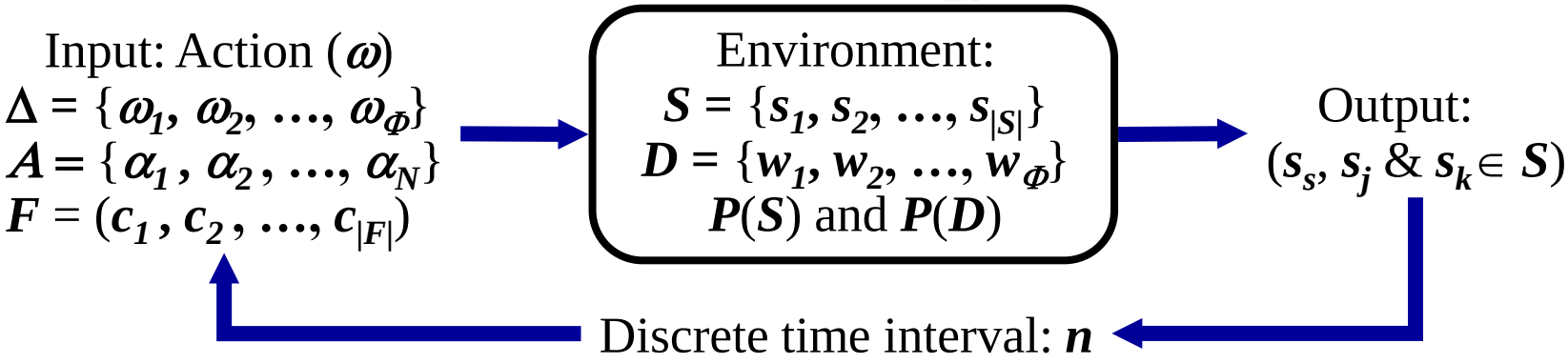
1. **Matching**: At interval n, match the n^{th} action $\omega_s(n)$ of the input string with the longest word in the dictionary, denoted as $w_s(n)$. $w_s(n) \in s_s(n)$.

2. **Merging**: two sets are randomly selected (say, s_j and s_k, $j \neq k \neq s$). Satellite sets take the identity of its parent singleton (Family Leader). The merger is preformed as follows:

   i. Both s_j and s_k are non-empty sets and with no satellites: replace s_j by the union set ($s_j \cup s_k$).

   ii. s_j is a singleton with satellite/s: remove $FBL(s_j)/2$ satellites from its block, s_k remain unchanged.

S&M algorithm: the finite case
-The Strategy-



iii $\underline{s_k \text{ is alone an empty-set or a singleton with satellite/s}}$: remove $FBL(S_k)/2$ satellites from its block, s_j remain unchanged.

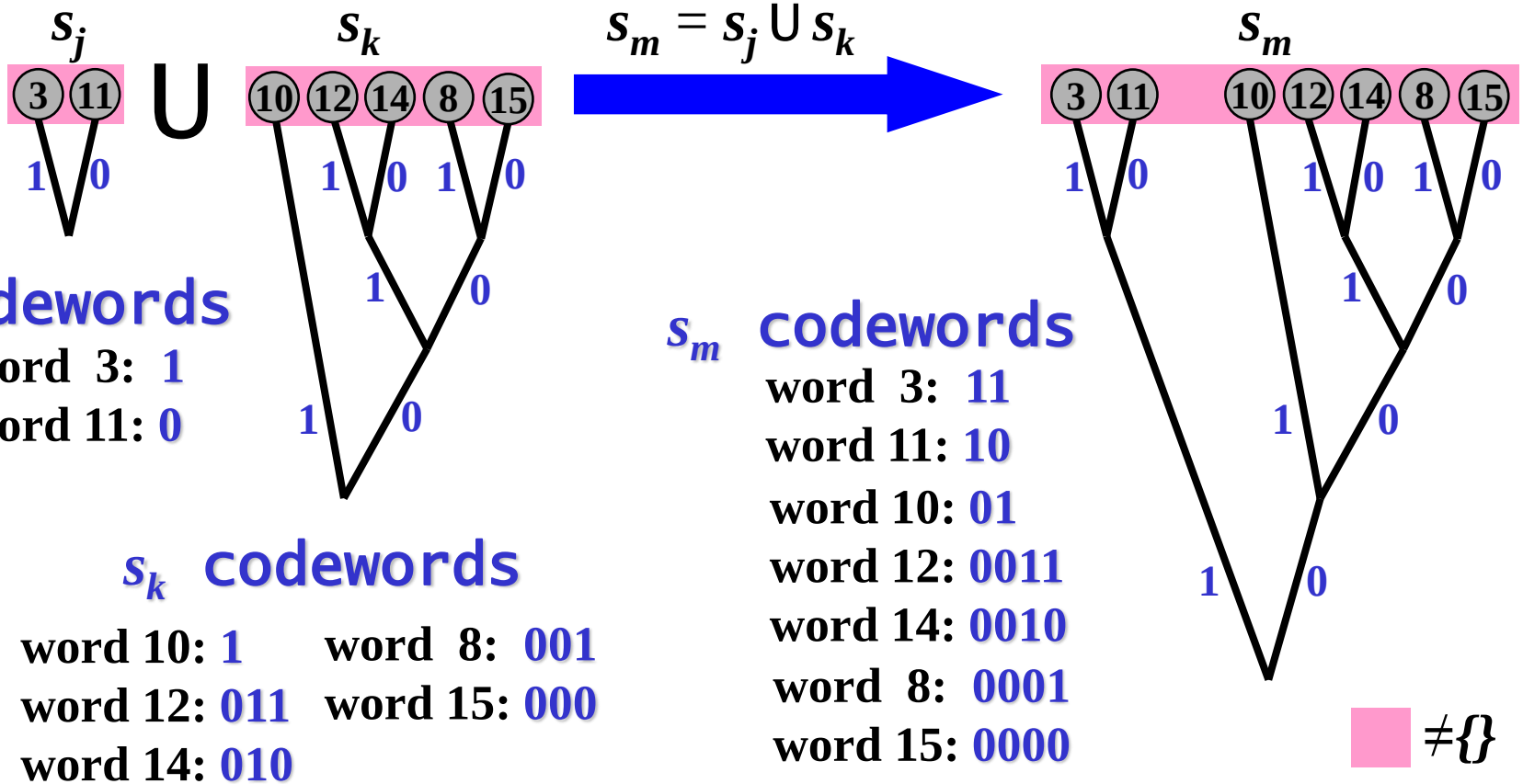
4 Splitting s_s:

i. $\underline{\text{Set size } |s_s| > 1}$: split s_s into two sets along its highest word link. (See diagram)

ii. $\underline{\text{Set } s_s \text{ is singleton with family block length } FBL(s_s)}$: add $FBL(s_s)$ satellite set . If $FBL(s_s)$ is greater than the available free set number, (free set number at the n^{th} interval is equal to the maximum number of sets minus the actual number of sets after the last merging step), then splitting procedure at this interval will be ignored.

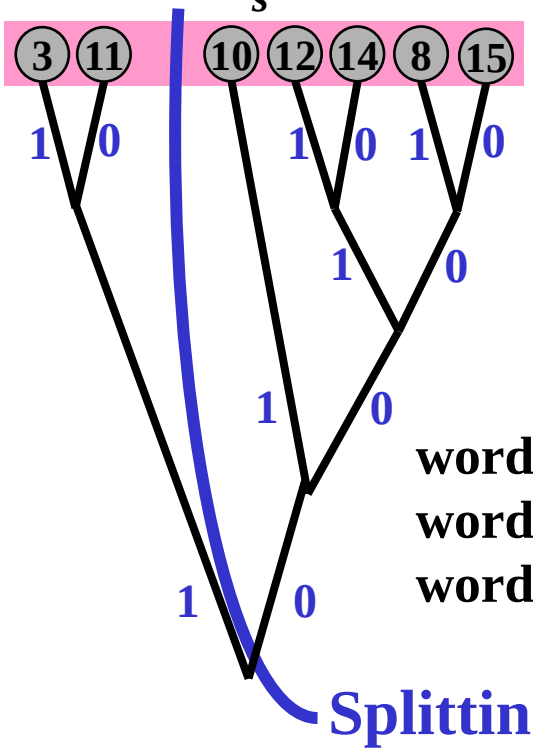
S&M algorithm: the finite case
-The Merger-

Non Empty Sets Merger: Merging two non-empty sets s_j and s_k into one set s_m, is a process of constructing the union of the two sets and updating the probabilities and codewords of all words within the set.



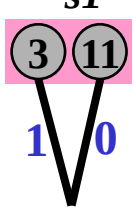
S&M algorithm: the finite case
-The Splitting-

Multi Word Sets Splitting: Splitting a multi word set s_s into two sets s_{s1} and set s_{s2}, along the highest splitting plane in the set, therefore most significant digit of s_s is removed, and the word probabilities been doubled.

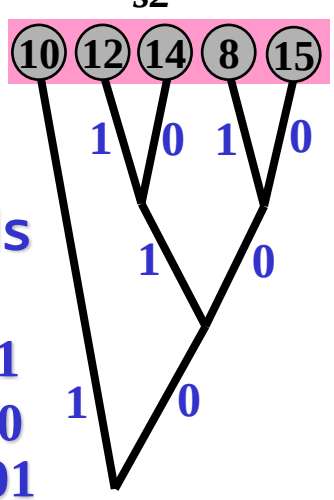


Splitting s_s into set s_{s1} & set s_{s2}

s_{s1} codewords
word 3: 1
word 11: 0



&



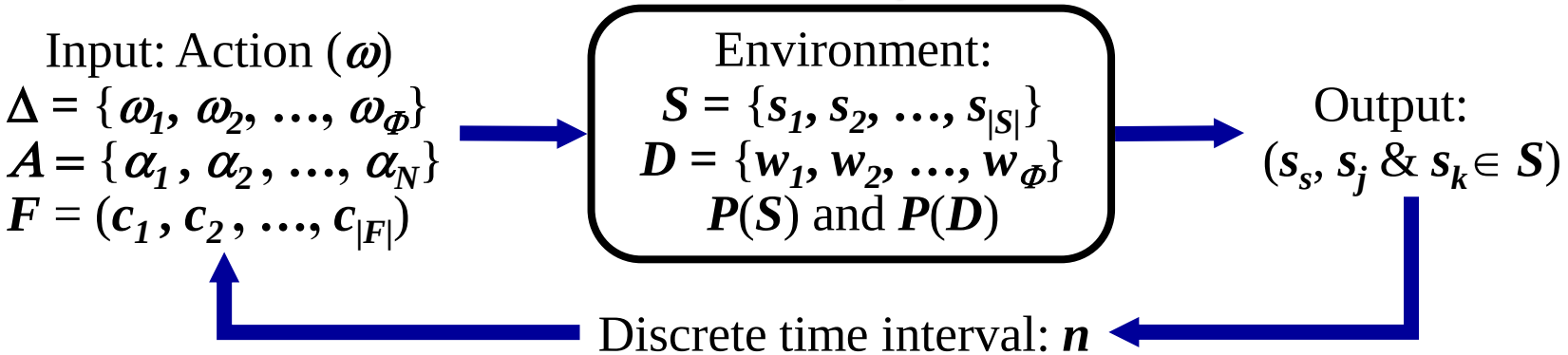
s_{s2} codewords
word 10: 1
word 12: 011
word 14: 010
word 8: 001
word 15: 000

s_s codewords

word 3: 11	word 12: 0011
word 11: 10	word 14: 0010
word 10: 01	word 8: 0001
	word 15: 0000

$\neq\{\}$

S&M algorithm: the finite case
-The Probability Vectors-



The probability vector of the source dictionary Δ is:

$$(p(\omega_1),\ p(\omega_2),\ \ldots,\ p(\omega_\Phi)).$$

The state probability vector of the environment dictionary D, is:

$$(p(w_1),\ p(w_2),\ \ldots,\ p(w_\Phi))$$

The real probability of the set s_i is given by: $p_{real}(s_i) = \sum_{j=1}^{|S_i|} p(\omega_{ij}) / FBL(s_i)$.

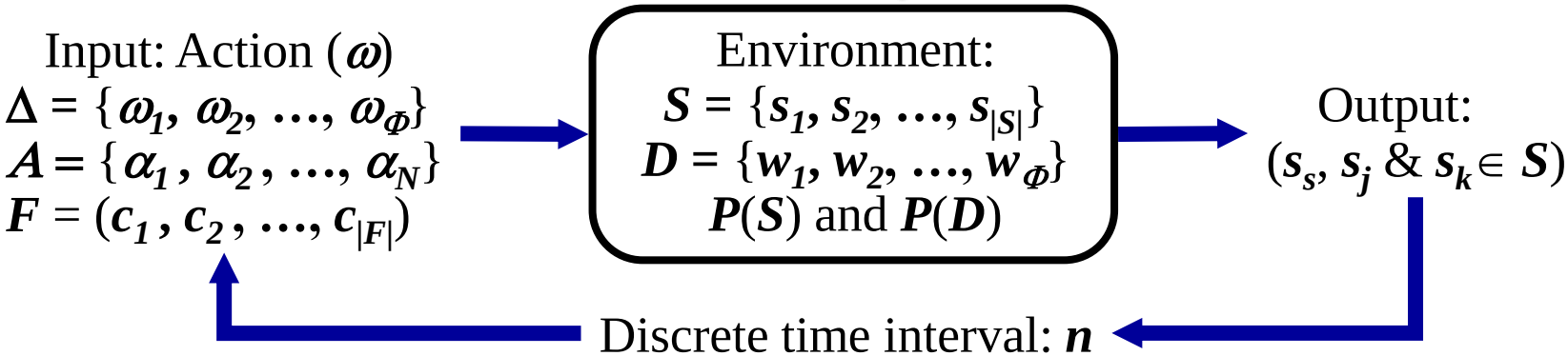
The state probability of the set s_i is given by: $p(s_i) = \sum_{j=1}^{|S_i|} p(w_{ij})$.

Set state probabilities $p(s_i)$ are equiprobable, equal to $(1 / |S|)$.

Since empty sets have zero frequency of accordance, their state probabilities, is therefore, added to their parent singleton.

Non empty set state probability is therefore: $p(s_i) = FBL(s_i) / |S|$.

S&M algorithm: the finite case
-The Probability Vectors-



The ***in-set*** probability vector $P_{in\text{-}set}$: is the state probabilities of words within a set s_i:

$$P_{in\text{-}set} = (p_{in\text{-}set}(w_{i1}),\ p_{in\text{-}set}(w_{i2}),\ \ldots,\ p_{in\text{-}set}(w_{i|s_i|})) \mid p_{in\text{-}set}(s_i) = \sum_{j=1}^{|S_i|} p_{in\text{-}set}(w_{ij}) = 1.$$

At the n^{th} interval the words ***in-set*** probabilities $p_{in\text{-}set}(w_i)$ are computed as follows:

1- Single word set: $p_{in\text{-}set}(w_1) = 1$.

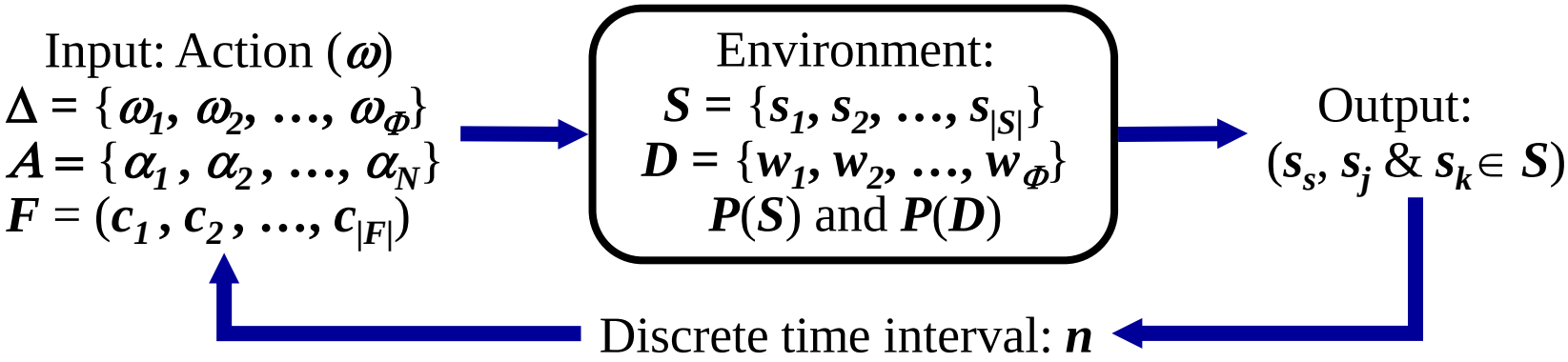
2- Two words set: $p_{in\text{-}set}(w_1) = p_{in\text{-}set}(w_2) = ½$.

3- More than two words in a set, their probabilities depend on the history of the set merger as shown in the diagram.

The state probability of a word $p(w_j) \in s_i$ is known as the ***out-set*** state probability vector given by the expression:

$$p(w_j) = \{ p_{in\text{-}set}(w_{ij}) \,.\, FBL(s_i) \ / |S| \mid w_j \in s_i \}.$$

S&M algorithm: the finite case
-Initial Conditions-



Initial State: When $n = 0$, the environment have the following initial conditions:

1) The dictionaries contains single character words only, $\Phi = N$.

2) Each sub-set of S contains a single word of the dictionary $\Delta \mid \omega_i \in s_i$. $|S| = \Phi$.

3) $p_{real}(\omega_i)$ is determined by the type of the input source.

4) Set family $FBL(s_i) = 1$, where $i = 1,2, \ldots, |S|$, $p(s_i) = 1/\Phi$.

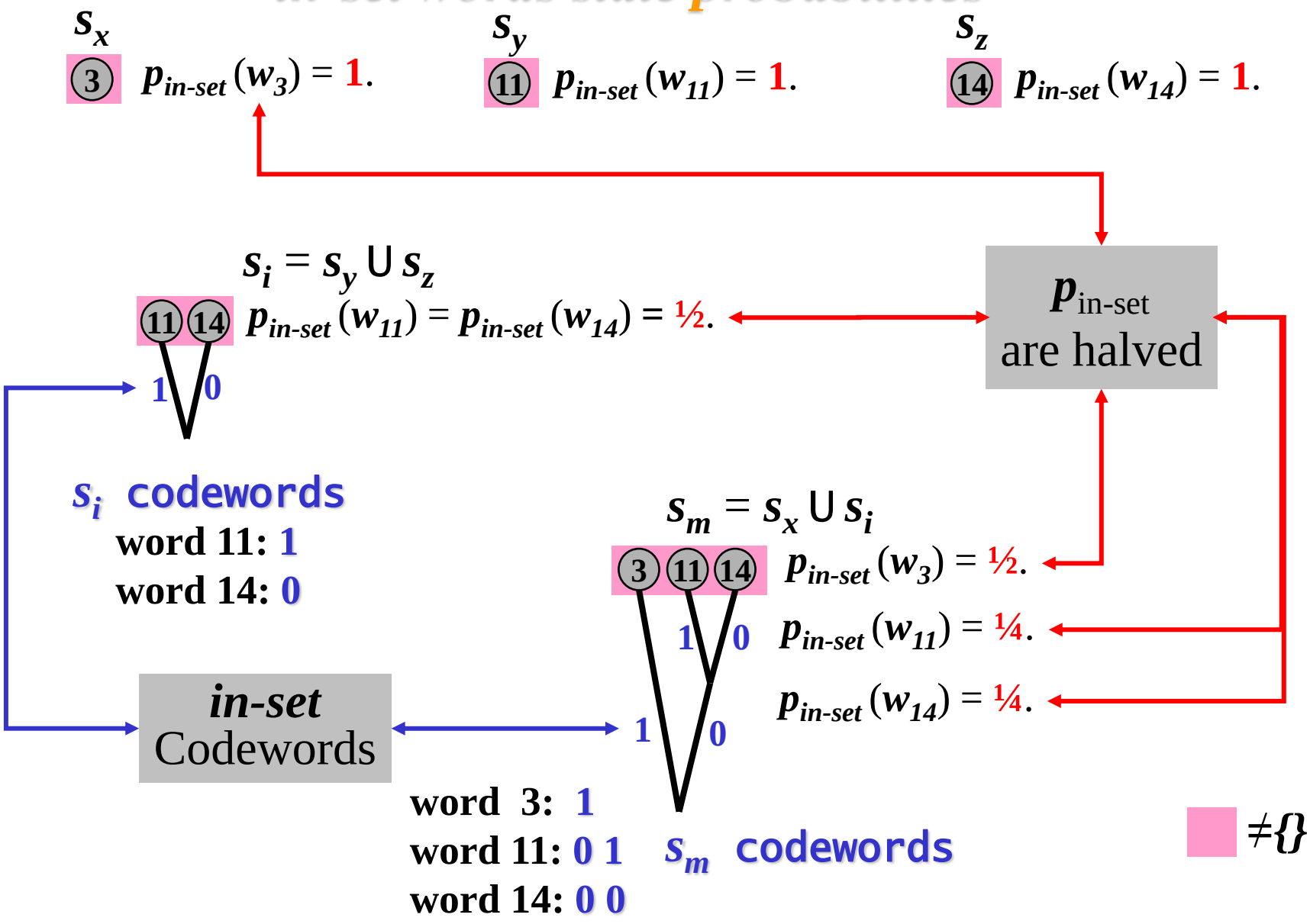
5) Since all words within the dictionaries at $n = 0$, all equiprobable, then:

$$Q_S(0) = 1 / N$$

$$H_S(0) = \log_2 (N)$$

$Q_W(0)$ and $H_W(0)$: are determined by the source statistical parameters.

S&M algorithm: the finite case
-in-set words state probabilities-



S&M algorithm: The Tree Structure

Set s_i: In the proposed ***S&M*** algorithm all nodes of ***EDT*** except the root node are partitioned into sets $s_1, s_2, \ldots, s_{|S|}$, for lossless ***S&M*** system $(\alpha + \chi) \leq |S|$ and usually $|S| = 2^r$, ***r*** a positive integer, while for lossy ***S&M*** system $|S| = 1, 2, \ldots$ depending on the given degradation factor. Each set of nodes s_i corresponds to a set of words W_i in ***ED***. All node sets s_i are mutually exclusive, (i.e. no node is in two sets). The union of the $|S|$sets is equal to the set of all nodes in the ***EDT*** excluding the root node. A set s_i may contains a single ***data-node***, multiple ***data-nodes*** or ***no-data node***. ***Singleton*** is a set with a single ***data-node***.

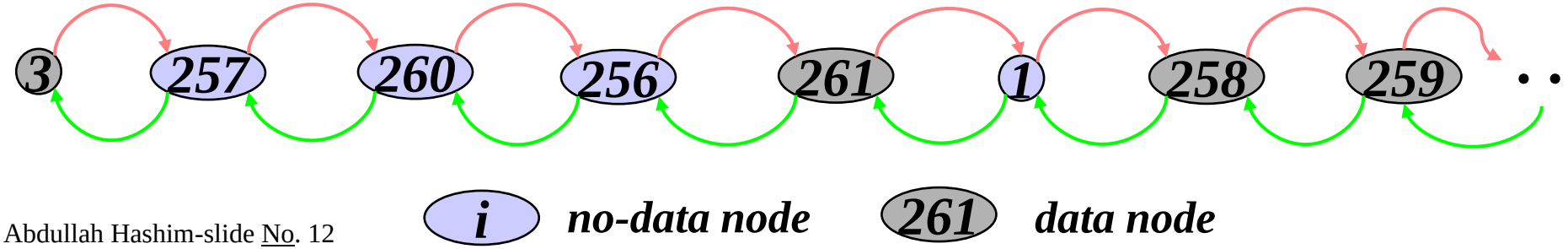
Null set: is a sets with ***no-data*** nodes may be called ***data-empty*** set and denoted by ***{}***.

Multiple data-nodes set: is a set with multiple data nodes, the set frequency of occurrence is equal to the sum of all the frequencies of occurrence of the node in the set.

S&M algorithm: The Tree Structure

Parent singleton set: is a one data-node set (s_i), leading a group of empty-sets called ***satellites*** of s_i. The number of satellites in the group called ***block length*** of s_i and denoted by $SBL(s_i)$ is equal to (2^r-1), (where ***r*** is a positive integer). The satellite block is located in an ordered sequence to the right of s_i in an ***OEDT.*** The last satellite in the block reside in an odd and the parent singleton in an even position of the ***OEDT***. $\{SBL(s_i) + 1\}$ is called the family block length of s_i and denoted by $\{FBL(s_i)\}$, where:
$1 \leq FBL(s_i) \leq (|S|/2)$; $FBL(s_i) = 2^r$, $r = 0, 1, 2, \ldots, \log_2(|S|/2)$.

Let nodes ***257***, ***260***, ***256***, and ***1*** to be ***no-data*** nodes of an ***OEDT***, then nodes ***257***, ***260*** and ***256*** are ***satellites*** of node ***3***, and said to be the ***satellite block*** of node ***3***, $SBL(s_3) = 3$. Node ***1*** is ***satellite*** of node ***261*** and are said to be the ***satellite block*** of node ***261***, $SBL(s_{261}) = 1$. Node ***3*** is said to be of higher ***rank*** than node ***261***.



S&M algorithm: The Tree Structure

Γ_i *is the* ***rank*** *of the* ***non-empty*** *node set* s_i *and its corresponding word set* W_i *in* ***D,*** *while the rank of* ***null satellite*** *node sets are equal to the* ***rank*** *of its parent* ***singleton*** *node* ***rank***.

The ***rank*** of set s_i is directly proportional to its ***satellite block*** length and inversely proportional to its ***set size***. The ***ranks*** of sets with same bock length and size is directly proportional to the maximum wordlength of W_i. It is a positive integer given by the expression:-

$$\Gamma_i = L_{max}(D) \,.\, (|D|_{\max} + SBL(s_i) - |s_i|) + L_{max}(W_i)$$

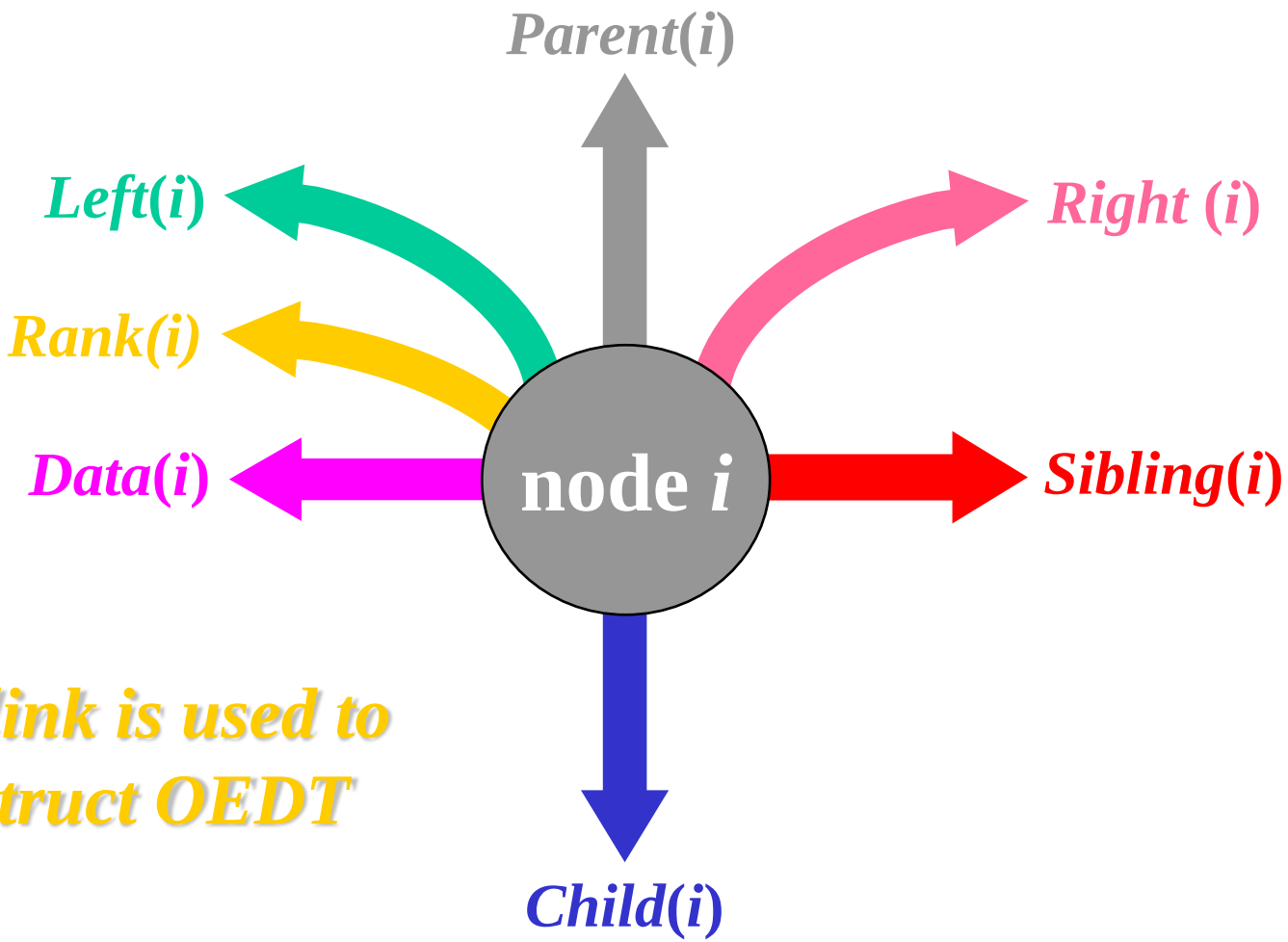
Note:

1. Since satellites have ***rank*** equal to that of its parent ***singleton***, therefore ***satellite block*** sets in ***ODT*** are always positioned immediately on the right of its parent ***singleton*** set.
2. ***Singleton*** sets are ordered in decreasing scale of their ***satellite block*** length and increasing scale of their ***size***.
3. Sets of the same ***satellite block*** length and ***size*** are ordered in decreasing scale of the largest wordlength in W_i; i.e. $L_{max}(W_i)$.

S&M algorithm: The Tree Structure

ODT links:
With rank link

If a link has a value equal -1, the link is a null link



Rank link is used to Construct OEDT

S&M algorithm: the finite case
-Word Coding-

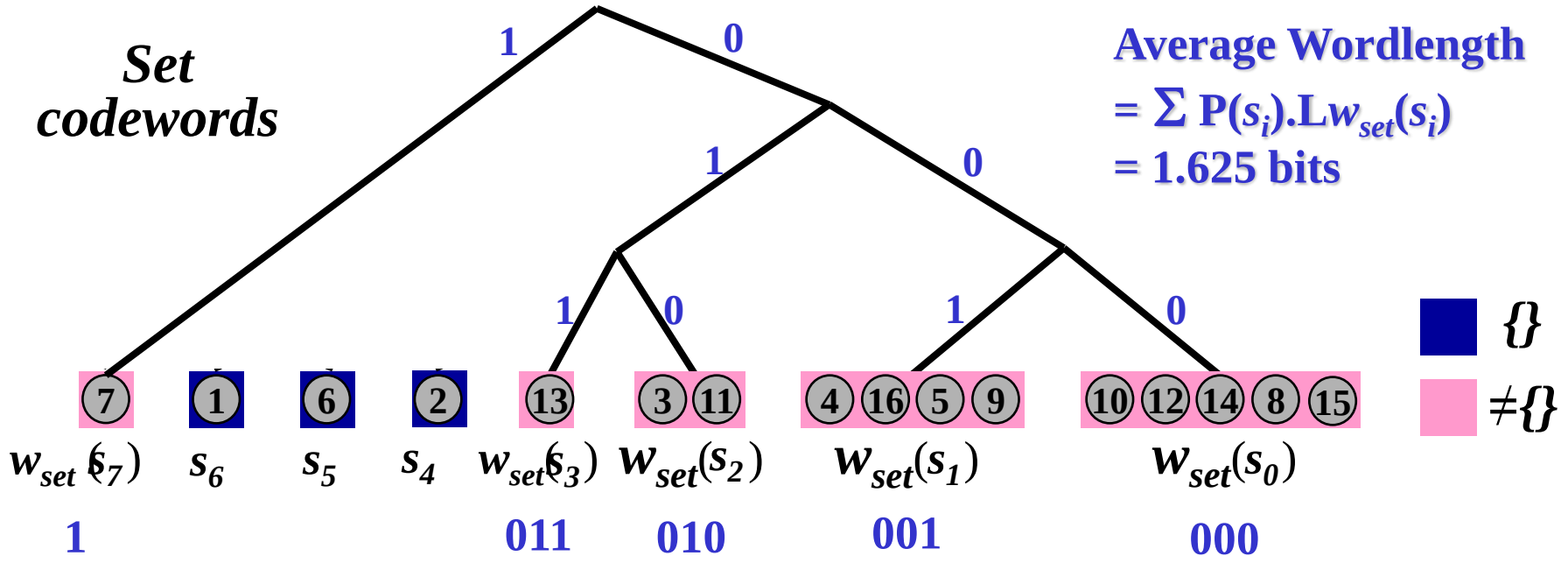
Set Coding:

Let ***D*** contains the following words:

$<w_7, w_1, w_6, w_2, w_{13}, w_3, w_{11}, w_4, w_{16}, w_5, w_9, w_{10}, w_{12}, w_{14}, w_8, w_{15}>$ partitioned into 8 sets: $(s_7, s_6, s_5, s_4, s_3, s_2, s_1, s_0)$; where:

$s_7 = \{w_7\}$; $s_6 = \{w_1\} = \{\}$; $s_5 = \{w_6\} = \{\}$; $s_4 = \{w_2\} = \{\}$; $s_3 = \{w_{13}\}$;

$s_2 = \{w_3, w_{11}\}$; $s_1 = \{w_4, w_{16}, w_5, w_9\}$; and. $s_0 = \{w_{10}, w_{12}, w_{14}, w_8, w_{15}\}$.



S&M algorithm: the finite case
-Words coding-

The word codeword

$w_{word} = w_{set}(s_i) + w_{inset}(v_i)$

String addition

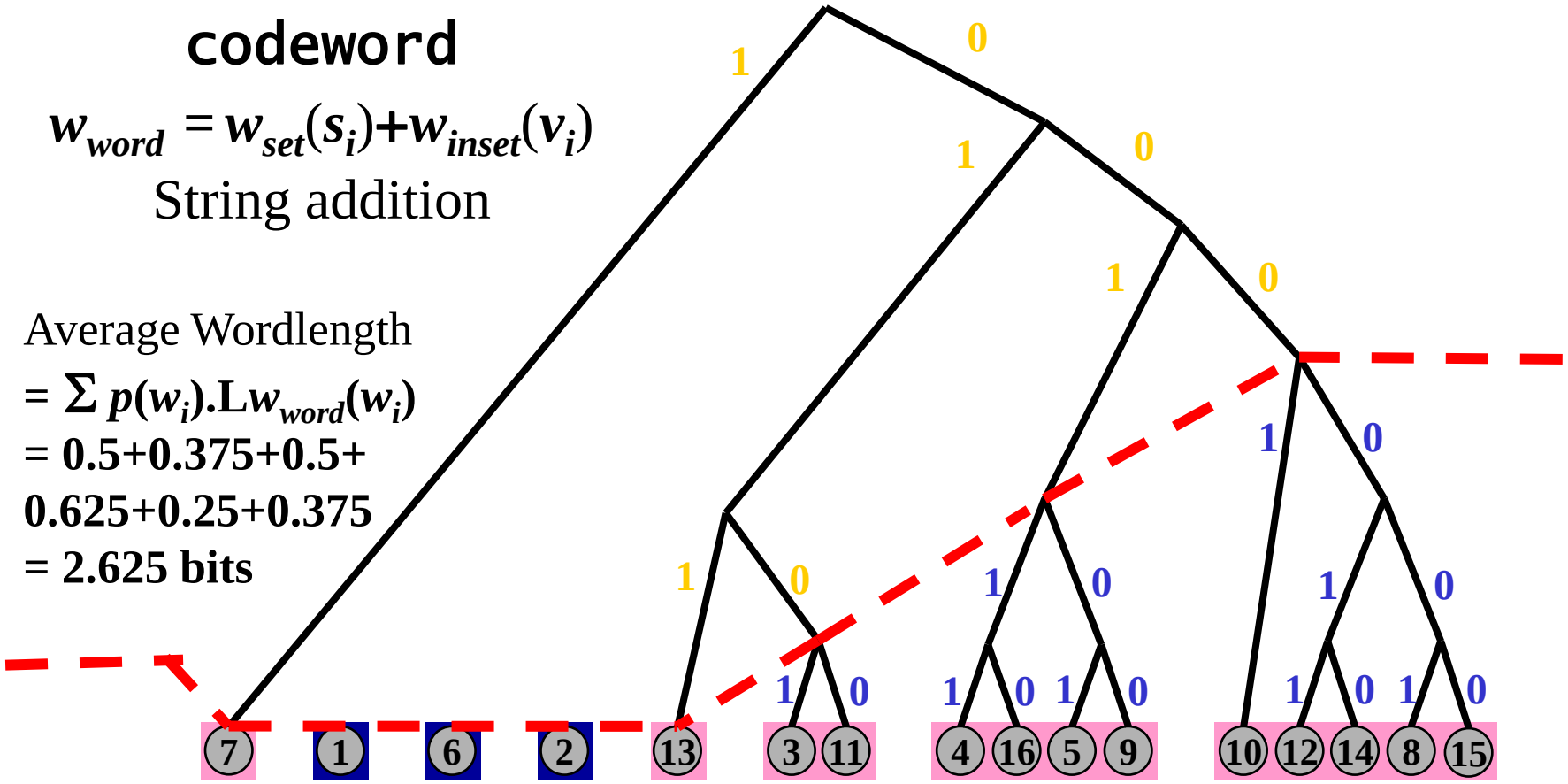
Average Wordlength

$= \sum p(w_i).Lw_{word}(w_i)$

= 0.5+0.375+0.5+

0.625+0.25+0.375

= 2.625 bits



s_7 s_6 s_5 s_4 s_3 s_2 s_1 s_0

word 7: 1
word 13: 011
word 3: 010 1
word 11: 011 0

word 4: 001 11
word 16: 001 10
word 5: 001 01
word 9: 001 00

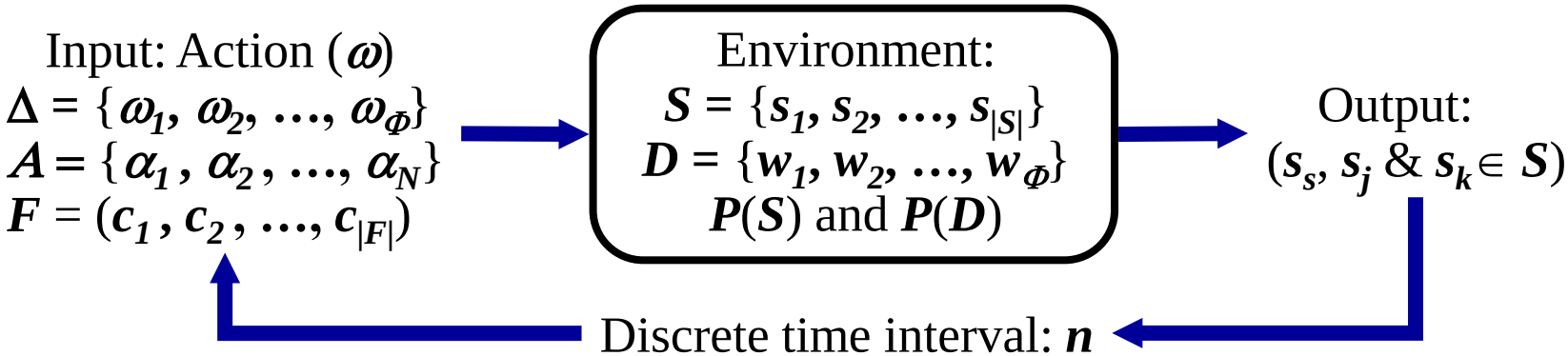
word 11: 000 1
word 12: 000 011
word 14: 000 010
word 8: 000 001

word 15: 000 000

{}

≠{}

S&M algorithm: the finite case
-The Practical Simulation-



Constants: In the finite case we assume:

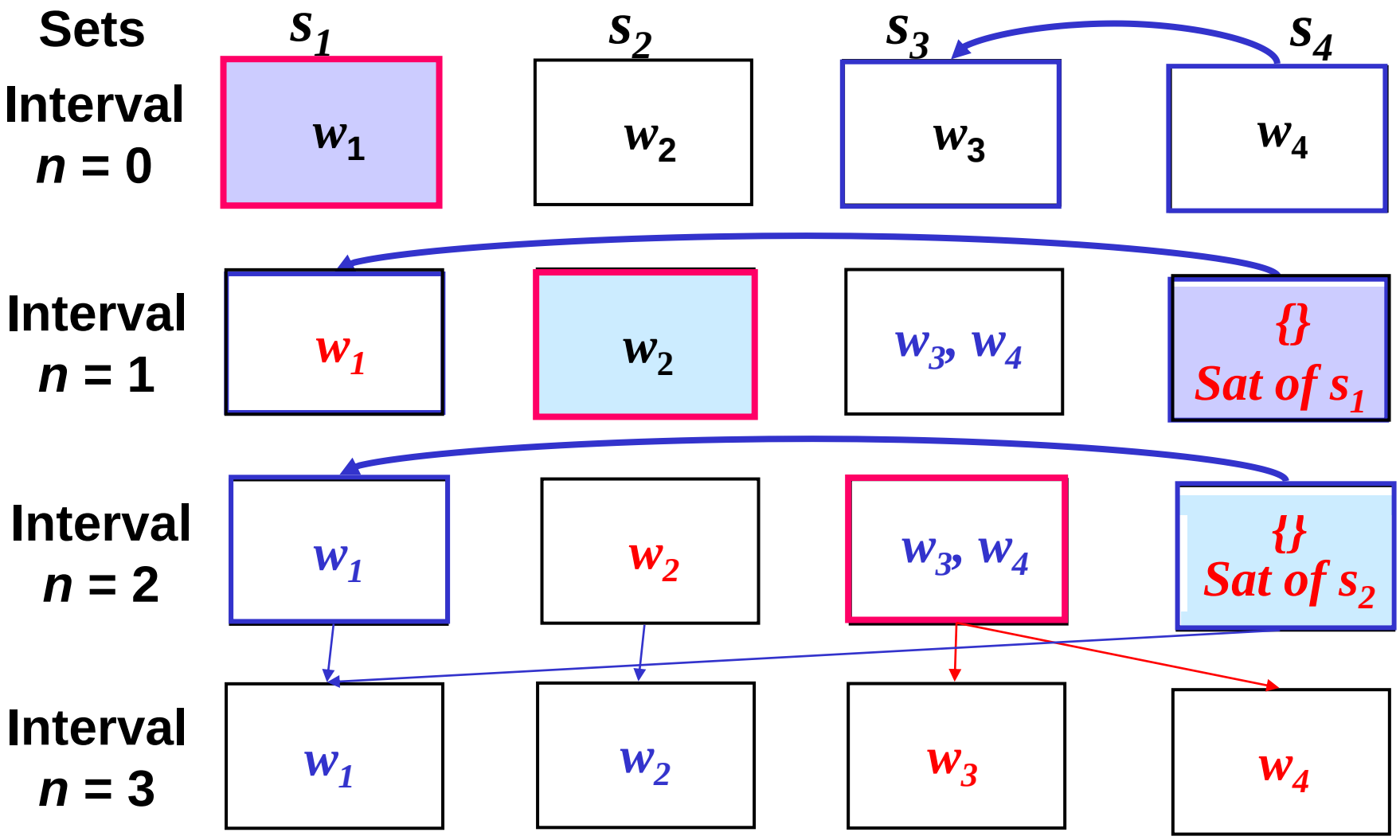
1) The word size in the dictionaries is one. i. e. $\Phi = N = 256$.
2) The algorithm tested for two sizes of sets: $|S| = 256$ and $|S| = 512$.
3) Two maximum number of intervals are used, $|F| = 256$ and 512.

Initial Conditions: The initial $p[s_1(0)]$ is set to one of ten values in the range of $\{0.001 \leq p[s_1(0)] \leq 1\}$ and all other set's probabilities is made to be equal to:

$$\{1 - p[s_1(0)]\} / (1 - |S|).$$

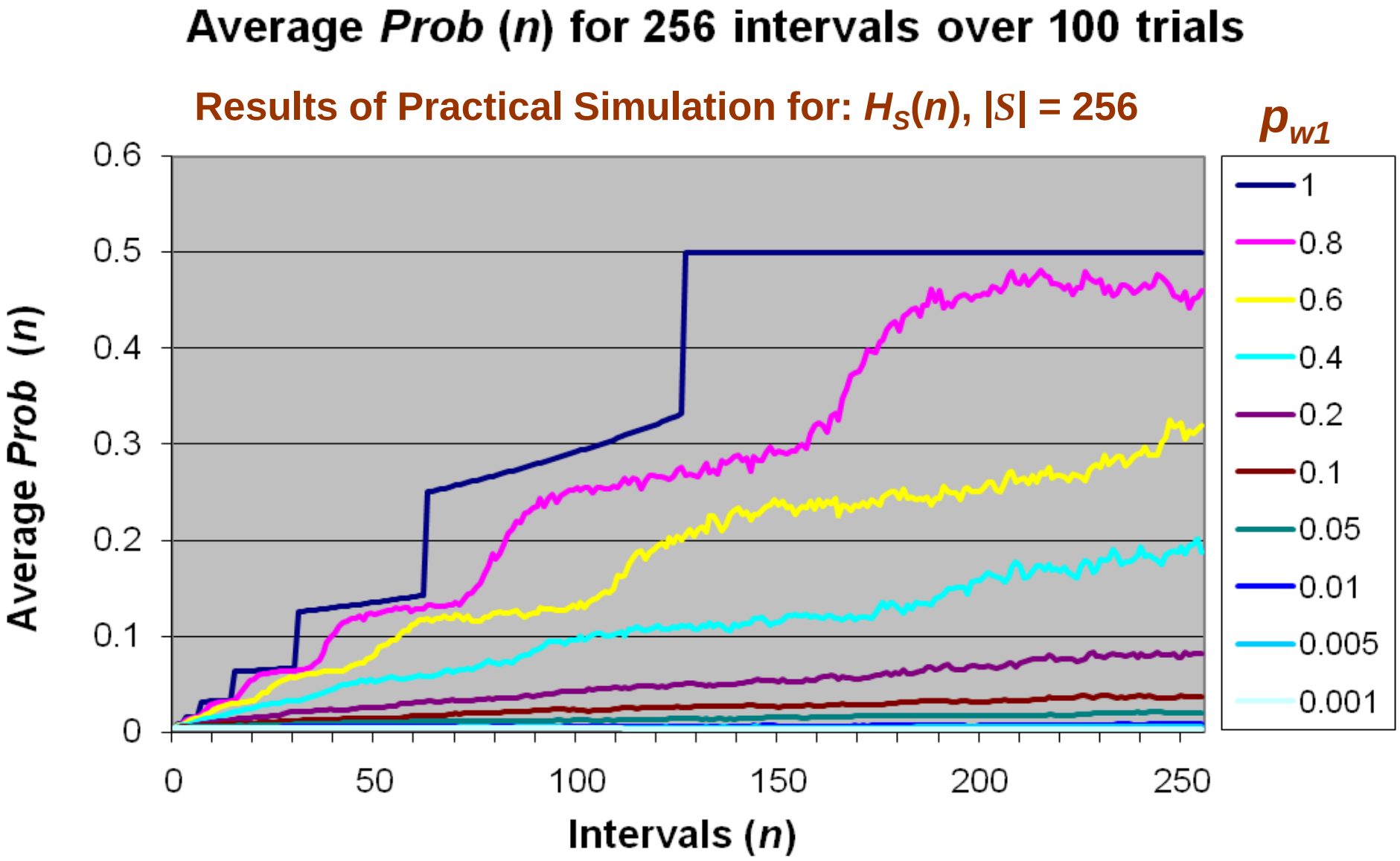
Results: The behaviour of the algorithm is determined by plotting the average values of $p(s_1(n))$, $Q_S(n)$, $H_S(n)$, $Q_D(n)$ and $H_D(n)$ over hundred **(100)** trials, for every one of the ten predetermined different set of initial probabilities.

S&M algorithm: the finite case

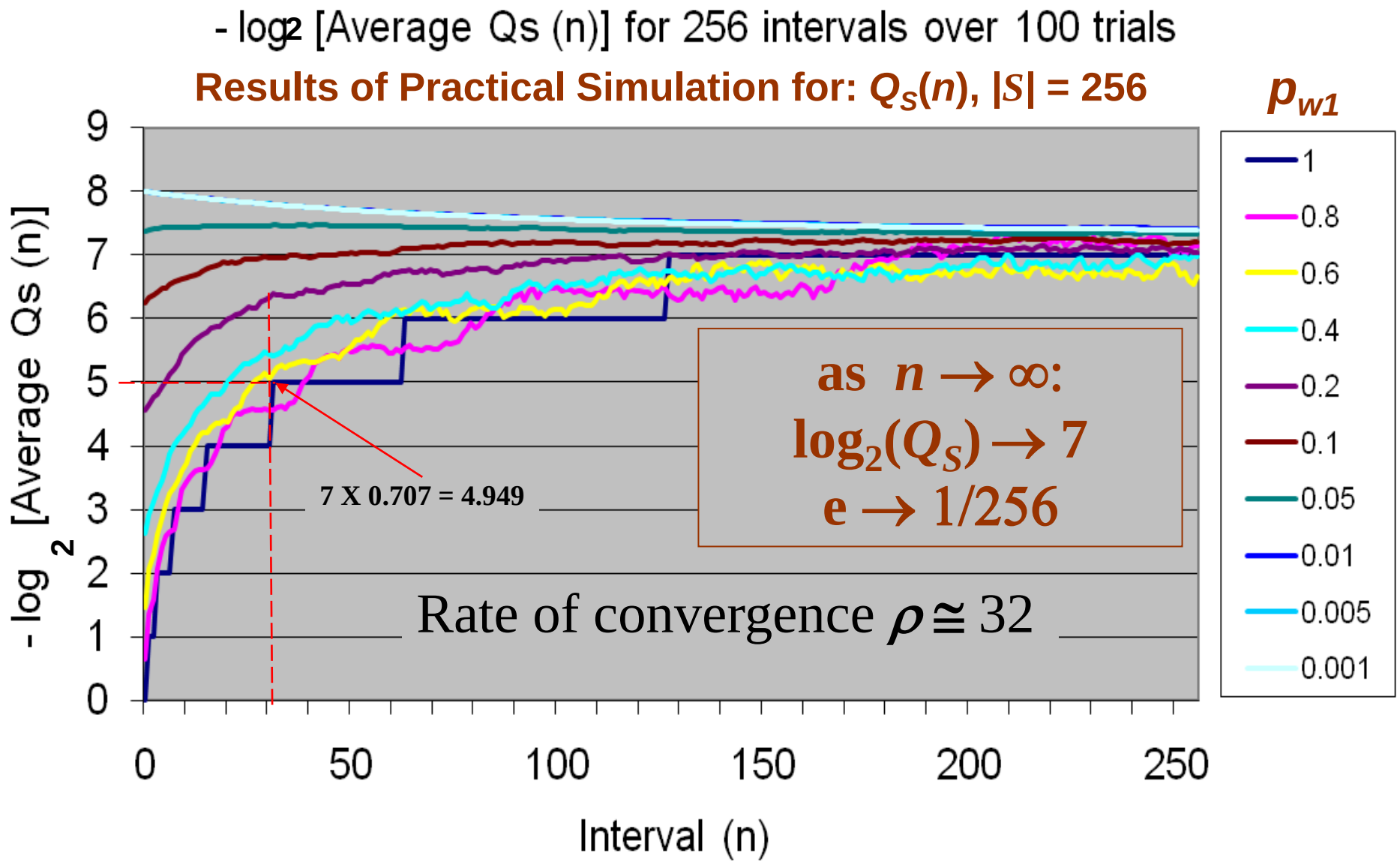


For large values of *n*, set probabilities will converge to $1/|S|$

S&M algorithm: the finite case
The S-Norms



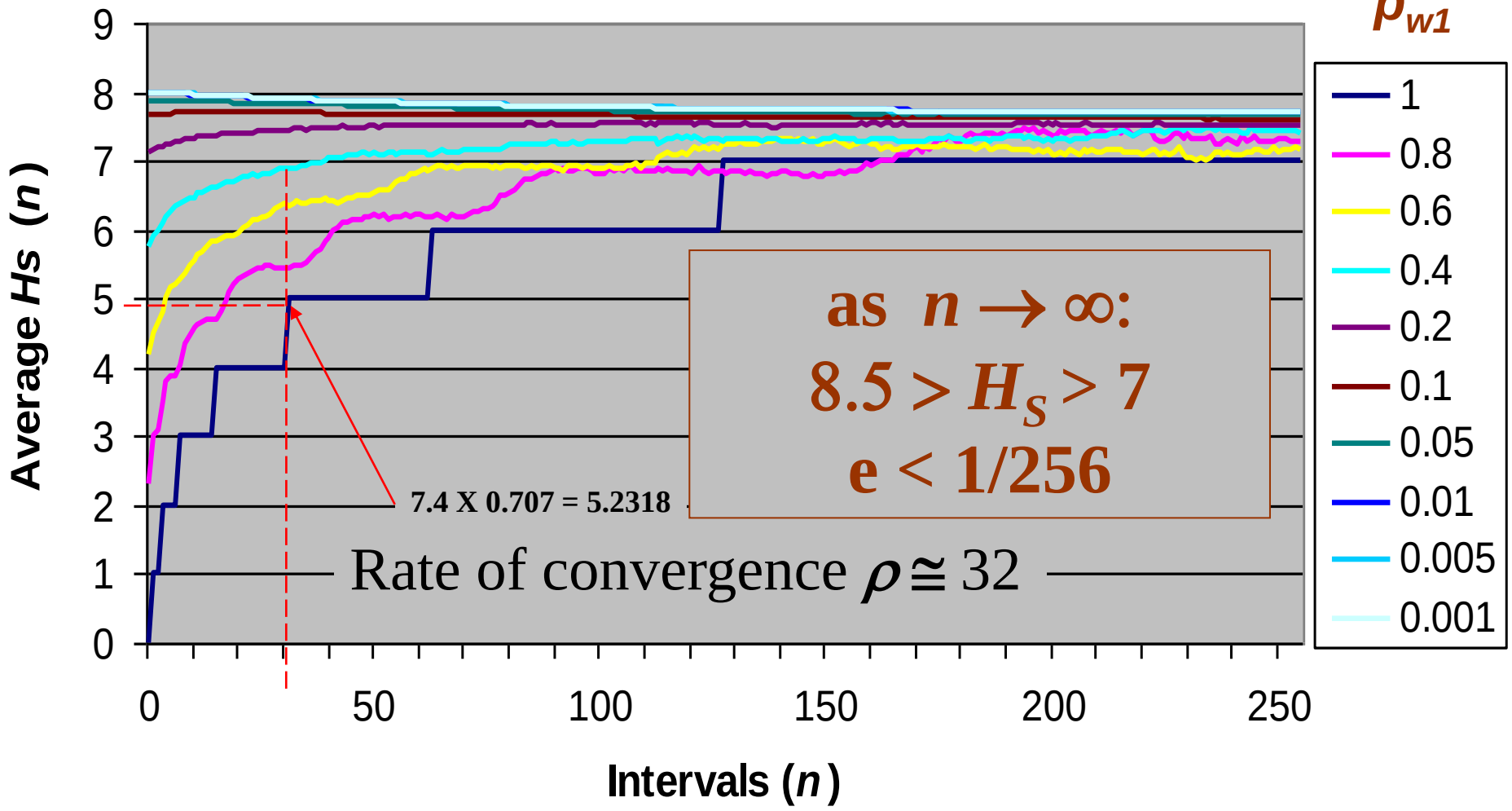
S&M algorithm: the finite case
The S-Norms



S&M algorithm: the finite case
The S-Norms

Average *Hs* (*n*) for 256 intervals over 100 trials

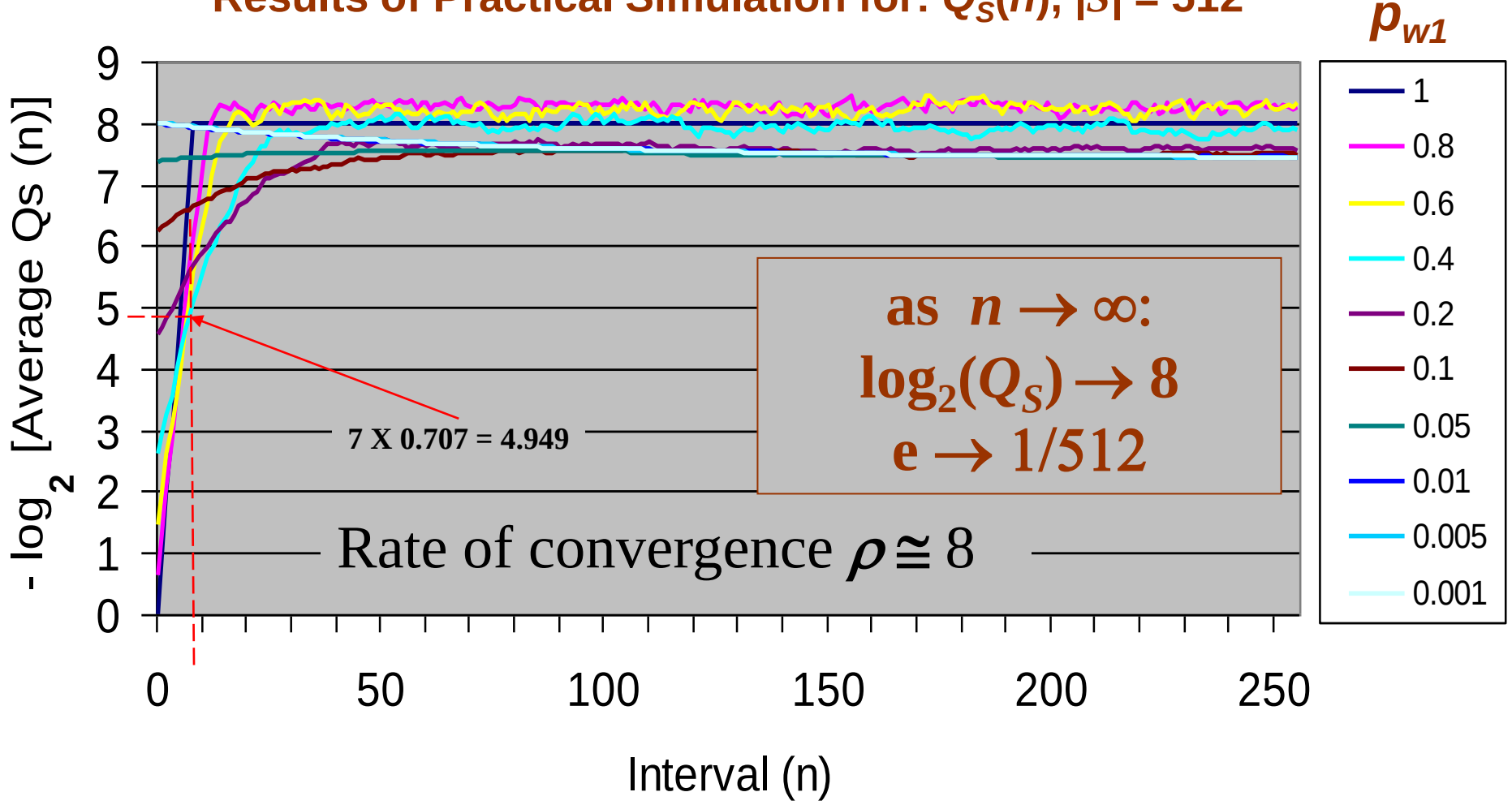
Results of Practical Simulation for: $H_S(n)$, $|S| = 256$



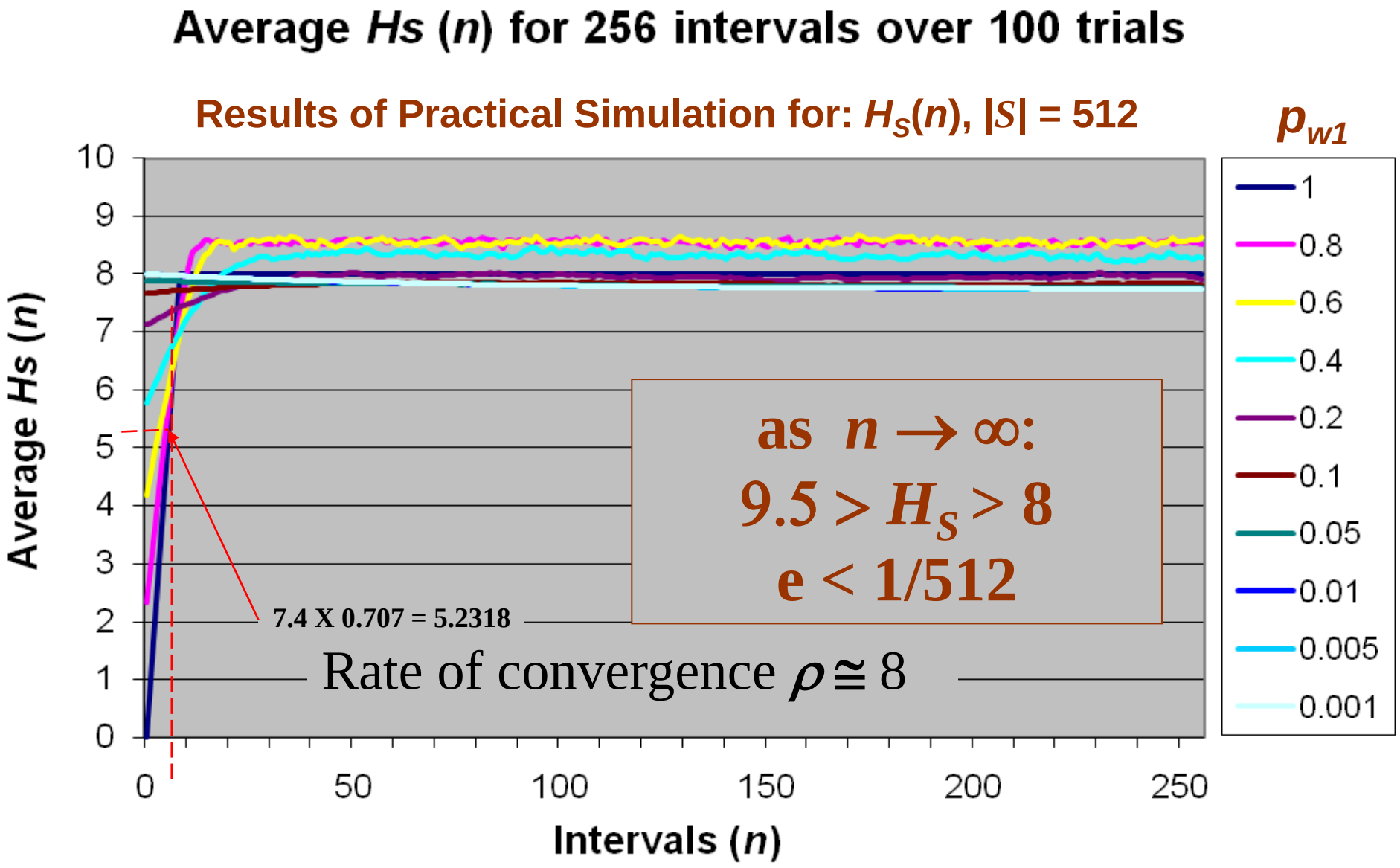
S&M algorithm: the finite case
The S-Norms

- $\log_2$[Average Qs (n)] for 256 intervals over 100 trials

Results of Practical Simulation for: $Q_S(n)$, $|S| = 512$



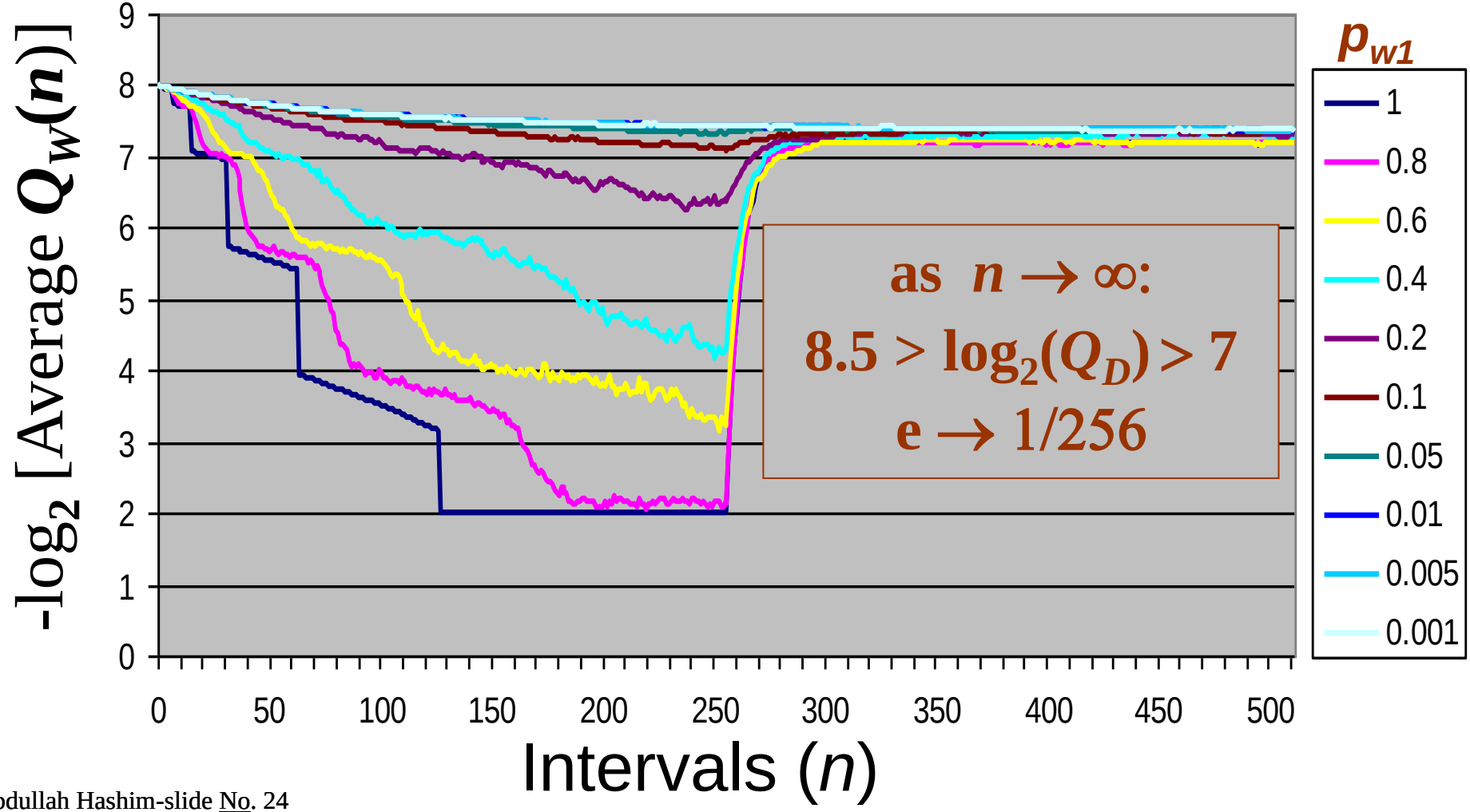
S&M algorithm: the finite case
The S-Norms



S&M algorithm: the finite case
The W-Norms

n = 0; Source probabilities are set in the given ten values.
At ***n*** = 256 the source become completely random.

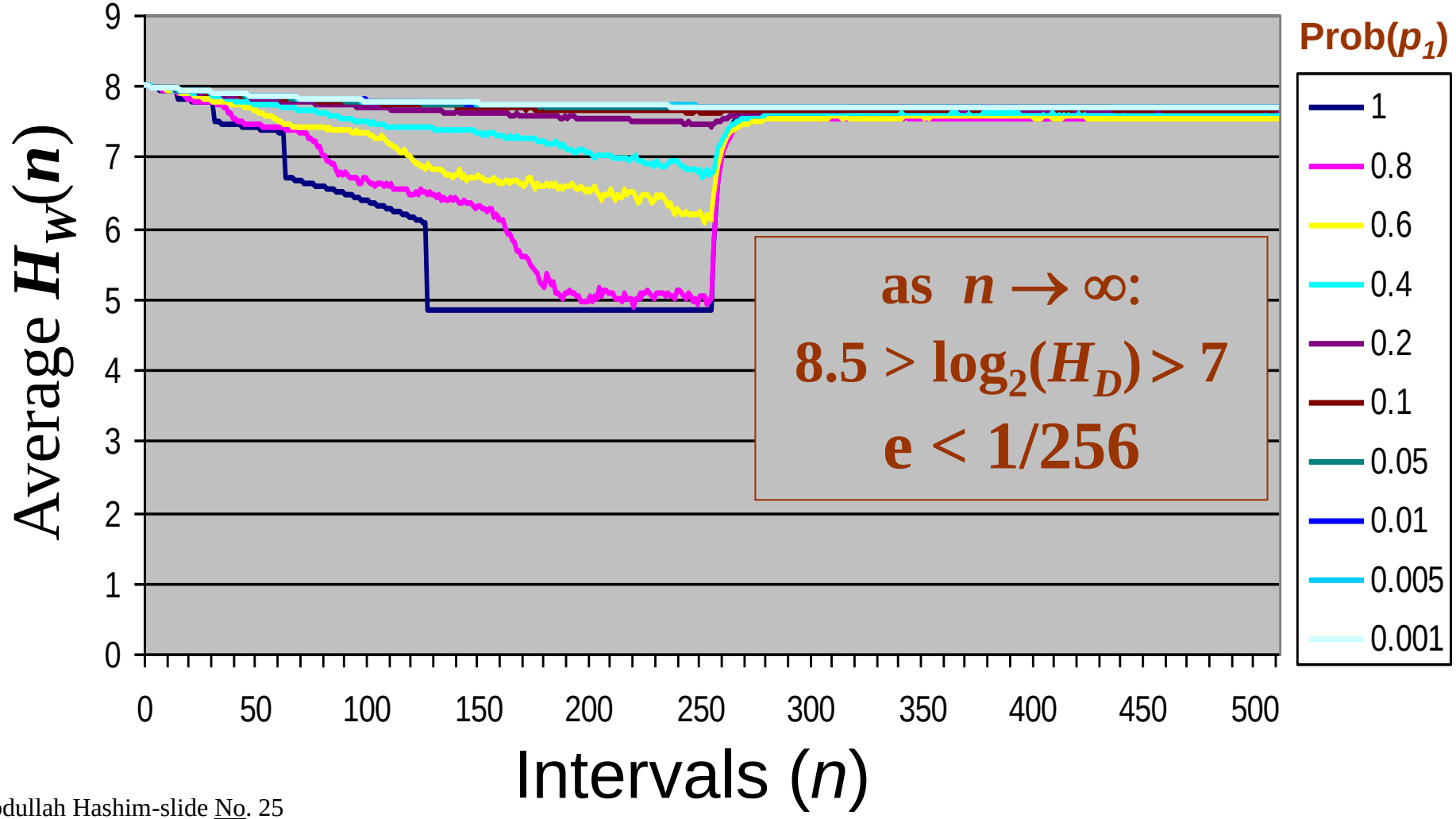
Results of Practical Simulation for: $Q_W(n)$, $|S| = 256$, over 100 trials



S&M algorithm: the finite case
The W-Norms

n = 0; Source probabilities are set in the given ten values.
At ***n*** = 256 the source become completely random.

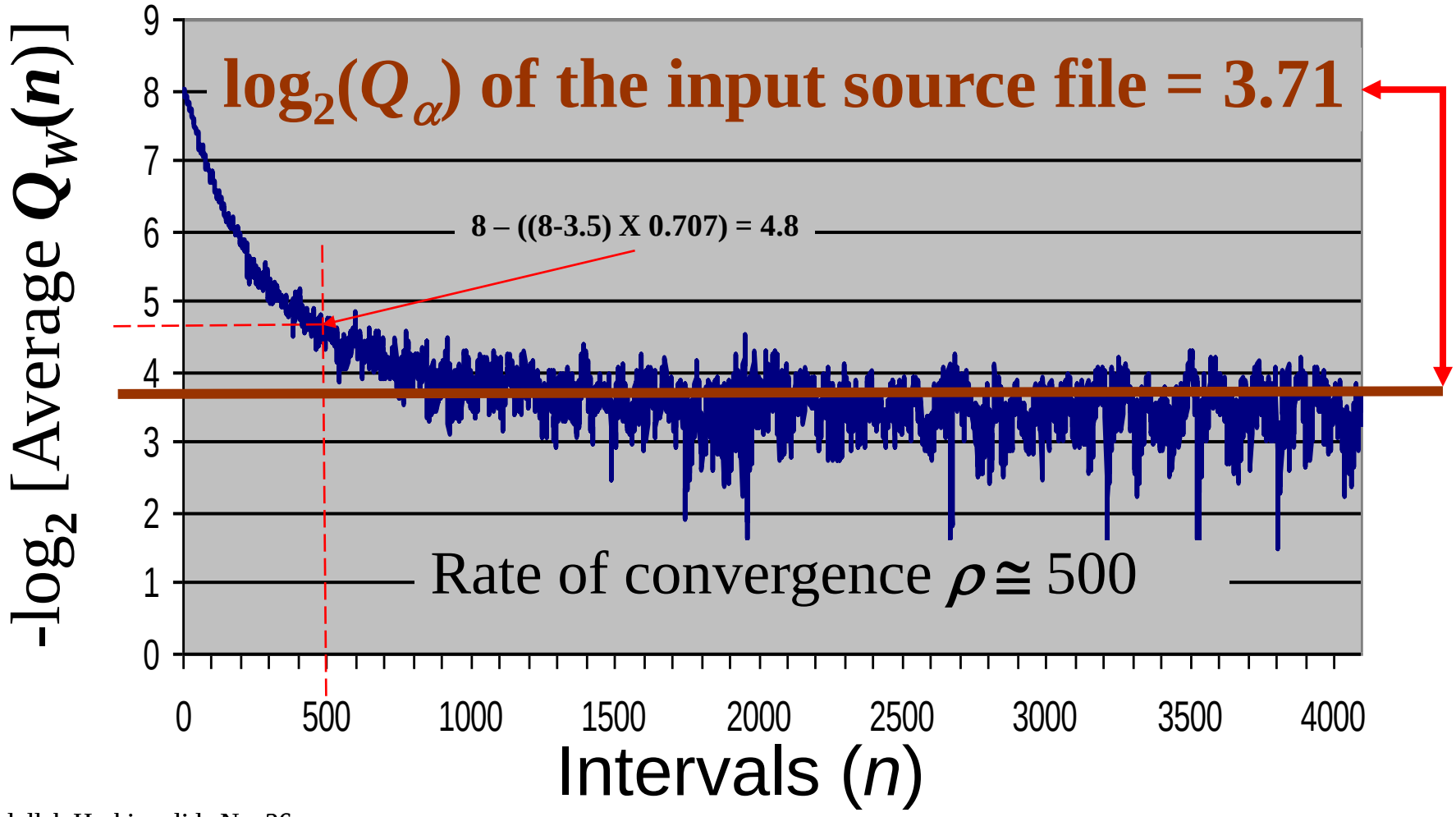
Results of Practical Simulation for: $H_W(n)$, $|S| = 256$, over 100 trials



S&M algorithm: the finite case
The W-Norms

Practical Input Source
4096 character string from Obama Inaugural Speech

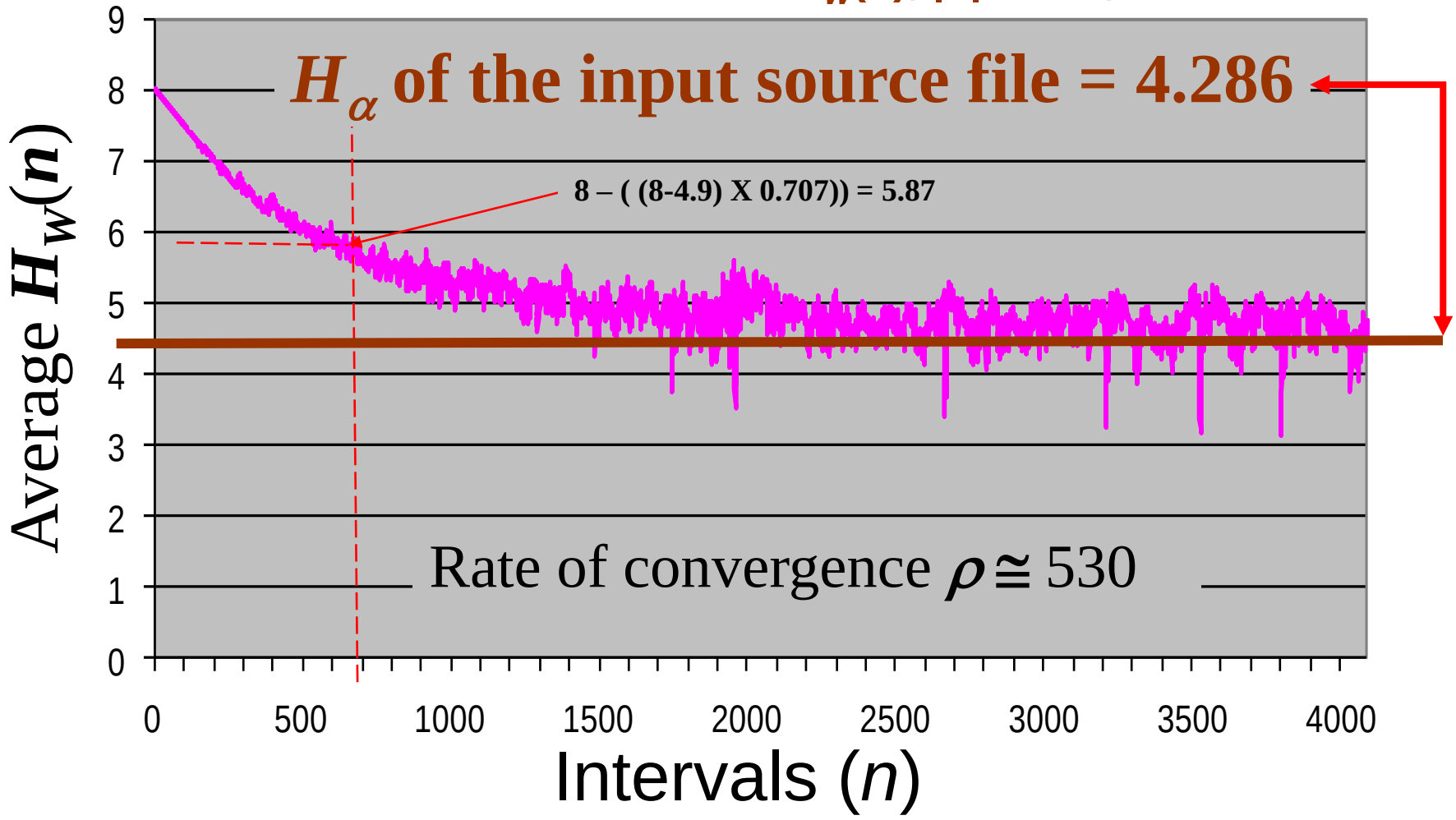
Results of Practical Simulation for: $Q_W(n)$, $|S| = 256$, over 100 trials



S&M algorithm: the finite case
The W-Norms

Practical Input Source
4096 character string from Obama Inaugural Speech

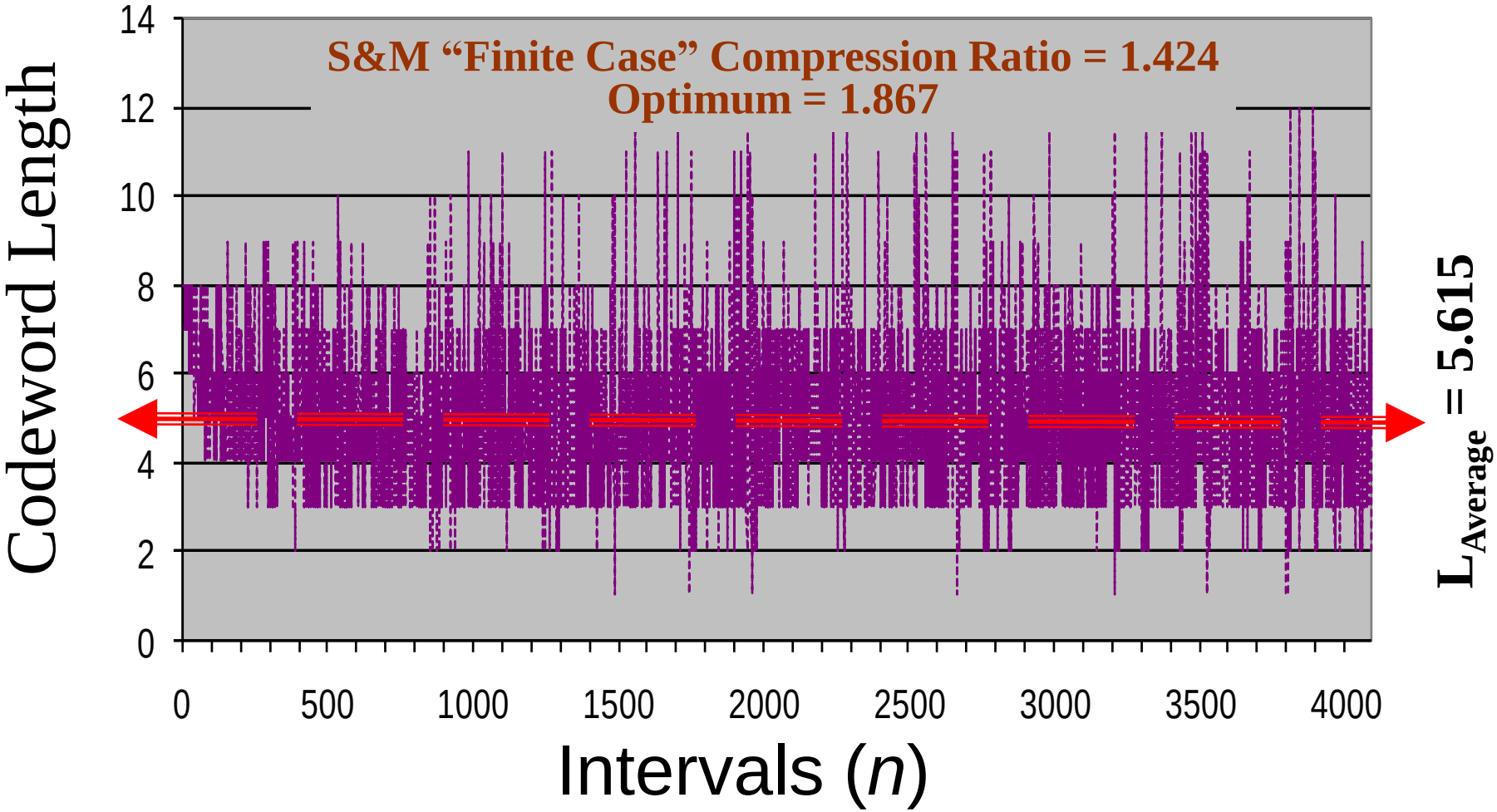
Results of Practical Simulation for: $H_W(n)$, $|S| = 256$, over 100 trials



S&M algorithm: the finite case
The D-Norms

Practical Input Source
4096 character string from Obama Inaugural Speech

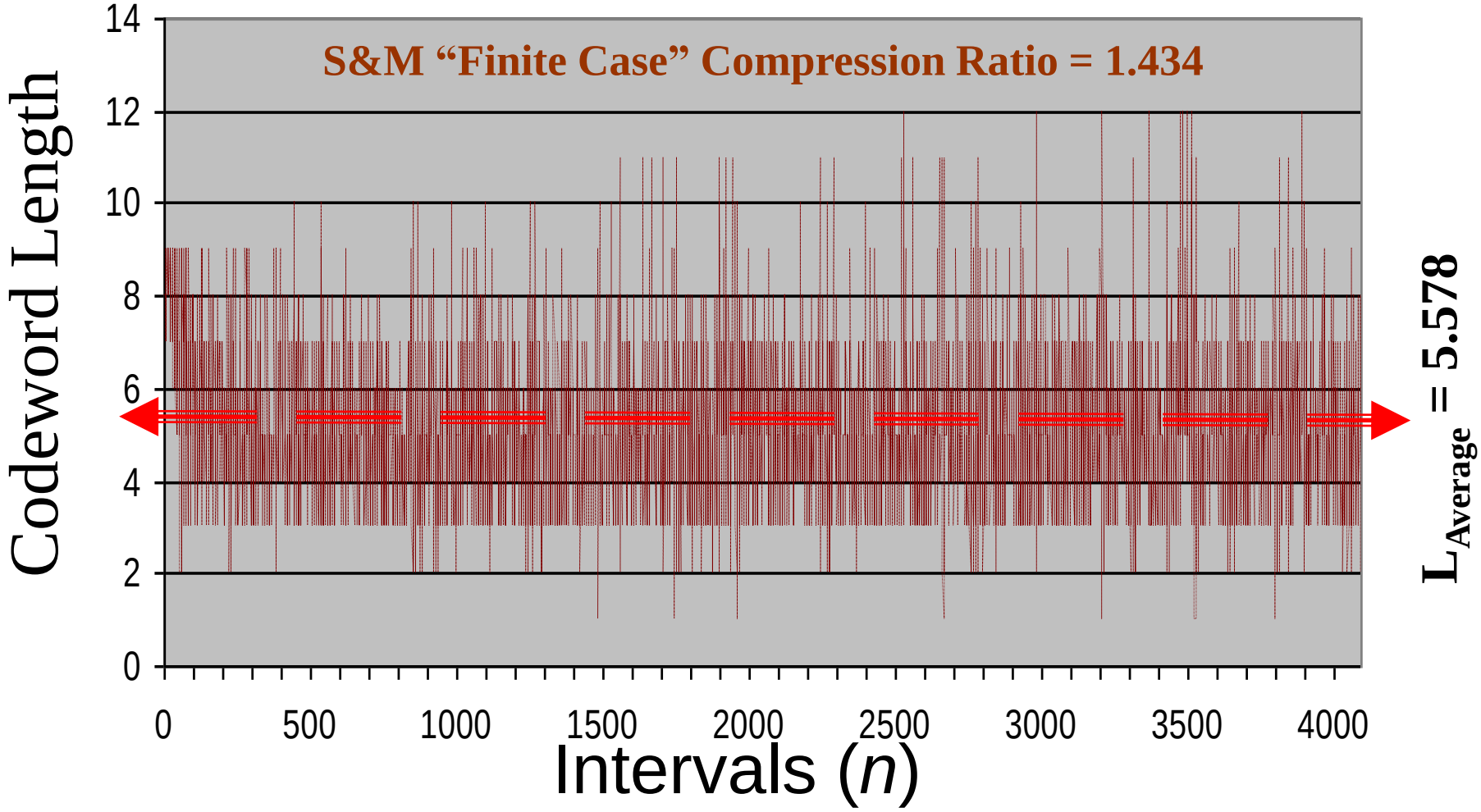
Results of Practical Simulation for: $|S| = 256$, over 100 trials



S&M algorithm: the finite case
The D-Norms

Practical Input Source
4096 character string from Obama Inaugural Speech

Results of Practical Simulation for: $|S| = 512$, over 100 trials



S&M algorithm: the finite case
The D-Norms

Practical Input Source
4096 character string from Obama Inaugural Speech

Results of Practical Simulation for: $|S| = 1024$, over 100 trials