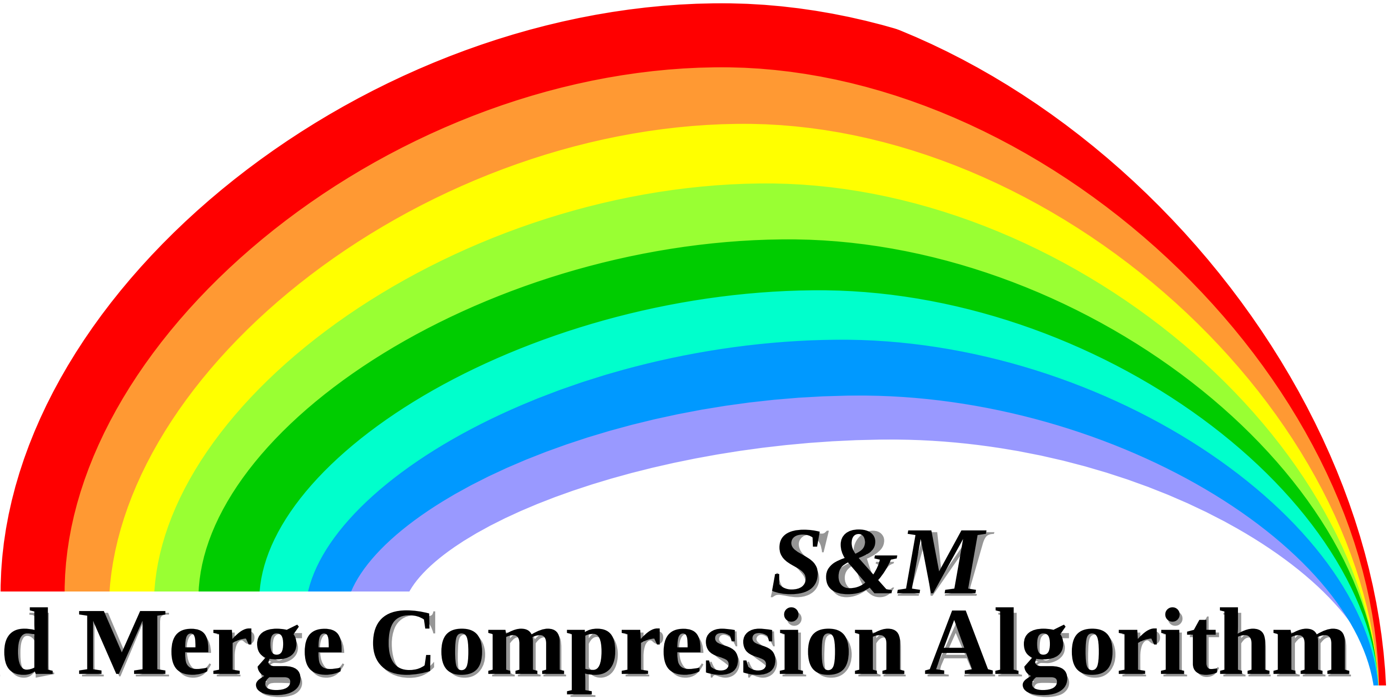


S&M

Split and Merge Compression Algorithm

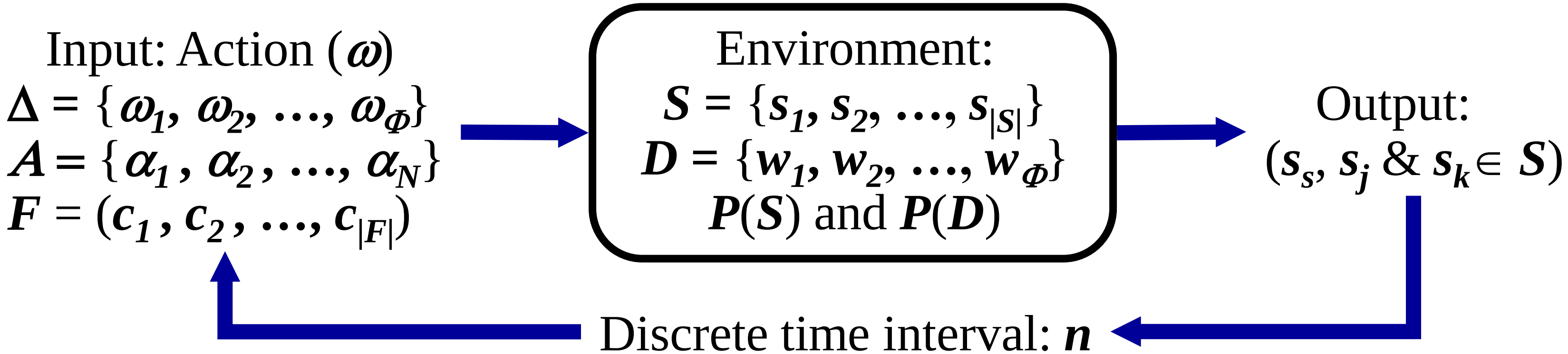
By

Abdullah Hashim

S&M algorithm: The Dictionary Case

S&M algorithm: the dictionary case

-The Concept-



Input: Action (ω)
$\Delta = \{\omega_1, \omega_2, \ldots, \omega_{\Phi}\}$
$A = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$
$F = (c_1, c_2, \ldots, c_{|F|})$

Environment:
$S = \{s_1, s_2, \ldots, s_{|S|}\}$
$D = \{w_1, w_2, \ldots, w_{\Phi}\}$
$P(S)$ and $P(D)$

Output:
(s_s, s_j & $s_k \in S$)

Discrete time interval: n

Assumptions: In the finite case we assume that: $|D| = |\Delta| = N$. In the dictionary case the size of the dictionary is allowed to grow. Asymptotically,

$$\lim_{\Phi \to \infty} H(w) \to H(\omega) \to 0.$$ *Lempel and Ziv, 1981.*

Practically, the size of the dictionary is limited to Φ, at this stage a word with lowest probability will be pruned to give a space for the new word to be added to the dictionary.

Dictionary Tree: The dictionary is best described and implemented through a tree data structure. Tree data structure offers a good foundation for understanding the behaviour and the practical implementation of the dictionary.

S&M algorithm: The Tree Structure

Dictionary tree (***DT***): A ***tree*** (***DT***) is an abstract data structure used here to represent the words in ***D***. It consists of a ***root*** node with ***links*** (branches) to its ***children*** nodes. These nodes in turn have links to their children. A ***leaf*** is a node with no children. The number of links in a path from the root to a node $\boldsymbol{v_i}$ is called the ***depth*** ($\boldsymbol{d_i}$) of the node $\boldsymbol{v_i}$. The maximum node depth of tree is called the ***tree depth*** ($\boldsymbol{d_t}$). The maximum number of nodes in a dictionary tree is ($\boldsymbol{\Phi}$=1). Each node of ***DT*** contains a single character from the alphabet, except that the root contains no data (i.e. contains the empty word $\boldsymbol{\Lambda}$).

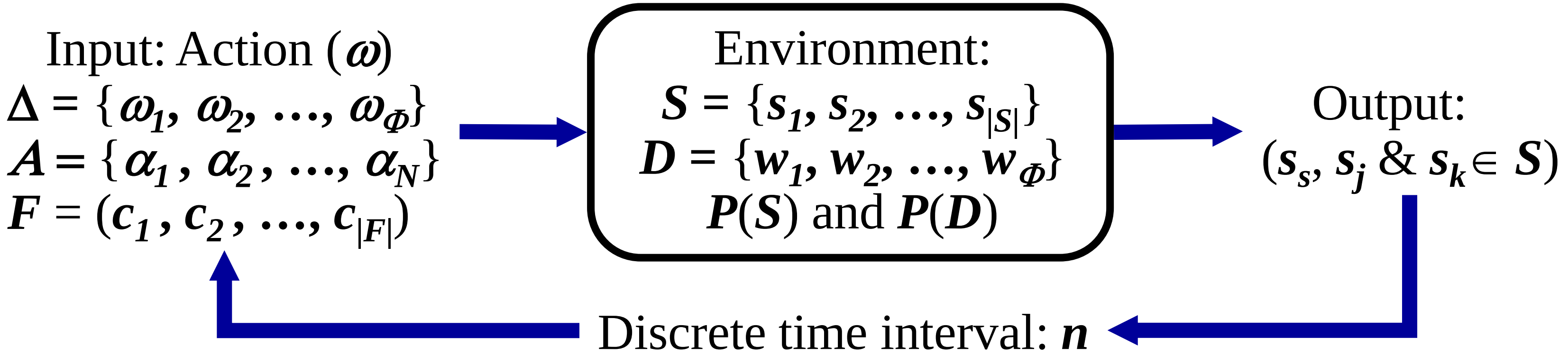
Data-node	A node containing a single character of the alphabet.
No-data node	A node containing the null character ($\boldsymbol{\Lambda}$).

The root of the ***DT*** has $\boldsymbol{\alpha}$ children. Each node $\boldsymbol{v_i}$, except the root node, corresponds to a word ($\boldsymbol{w_i}$) in ***D***.

The corresponding word of a given node may be constructed by reading the data $\boldsymbol{\Lambda}$ of the root node $\boldsymbol{v_0}$ and concatenated to the single character data from the ancestor nodes along the path in sequence, terminating with the character of the given node.

S&M algorithm: the dictionary case

-The Strategy-



Input: Action (ω)
$\Delta = \{\omega_1, \omega_2, \ldots, \omega_\Phi\}$
$A = \{\alpha_1 , \alpha_2 , \ldots, \alpha_N\}$
$F = (c_1 , c_2 , \ldots, c_{|F|})$

Environment:
$S = \{s_1, s_2, \ldots, s_{|S|}\}$
$D = \{w_1, w_2, \ldots, w_\Phi\}$
$P(S)$ and $P(D)$

Output:
$(s_s, s_j \;\&\; s_k \in S)$

Discrete time interval: n

The new word w_{new}: Let word ω_n , be an input string from file F, at interval n, match ω_n to the longest word w_n in the dictionary D:

$$\omega_n = w_n = <\alpha_1\alpha_2\ldots\alpha_i \ldots\alpha_j>.$$

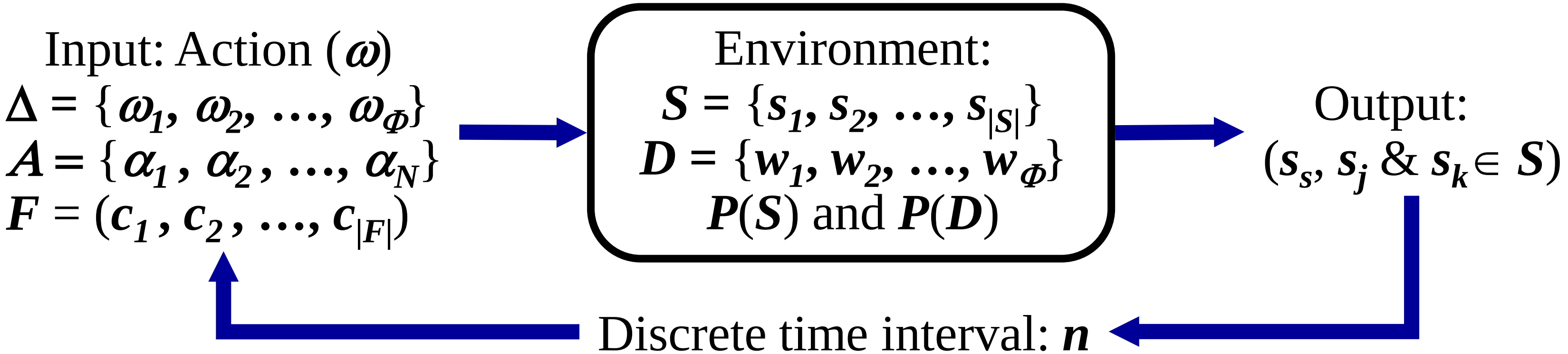
If α_{j+1} is the unmatched character resulting from the matching process, then the new word (w_{new}) is defend as:

$$w_{new} = <a_1a_2\ldots a_j a_{j+1}>.$$

The strategy of the dictionary case is identical to that, of the finite case, however, a singleton set is splitted by adding a new singleton set containing the new word (w_{new}). If (w_{new}) does not exist, then the process will be identical to that, of the finite case.

S&M algorithm: the finite case

-The Probability Vectors-



Input: Action (ω)

$\Delta = \{\omega_1, \omega_2, \ldots, \omega_\Phi\}$

$A = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$

$F = (c_1, c_2, \ldots, c_{|F|})$

Environment:

$S = \{s_1, s_2, \ldots, s_{|S|}\}$

$D = \{w_1, w_2, \ldots, w_\Phi\}$

$P(S)$ and $P(D)$

Output:

$(s_s, s_j \;\&\; s_k \in S)$

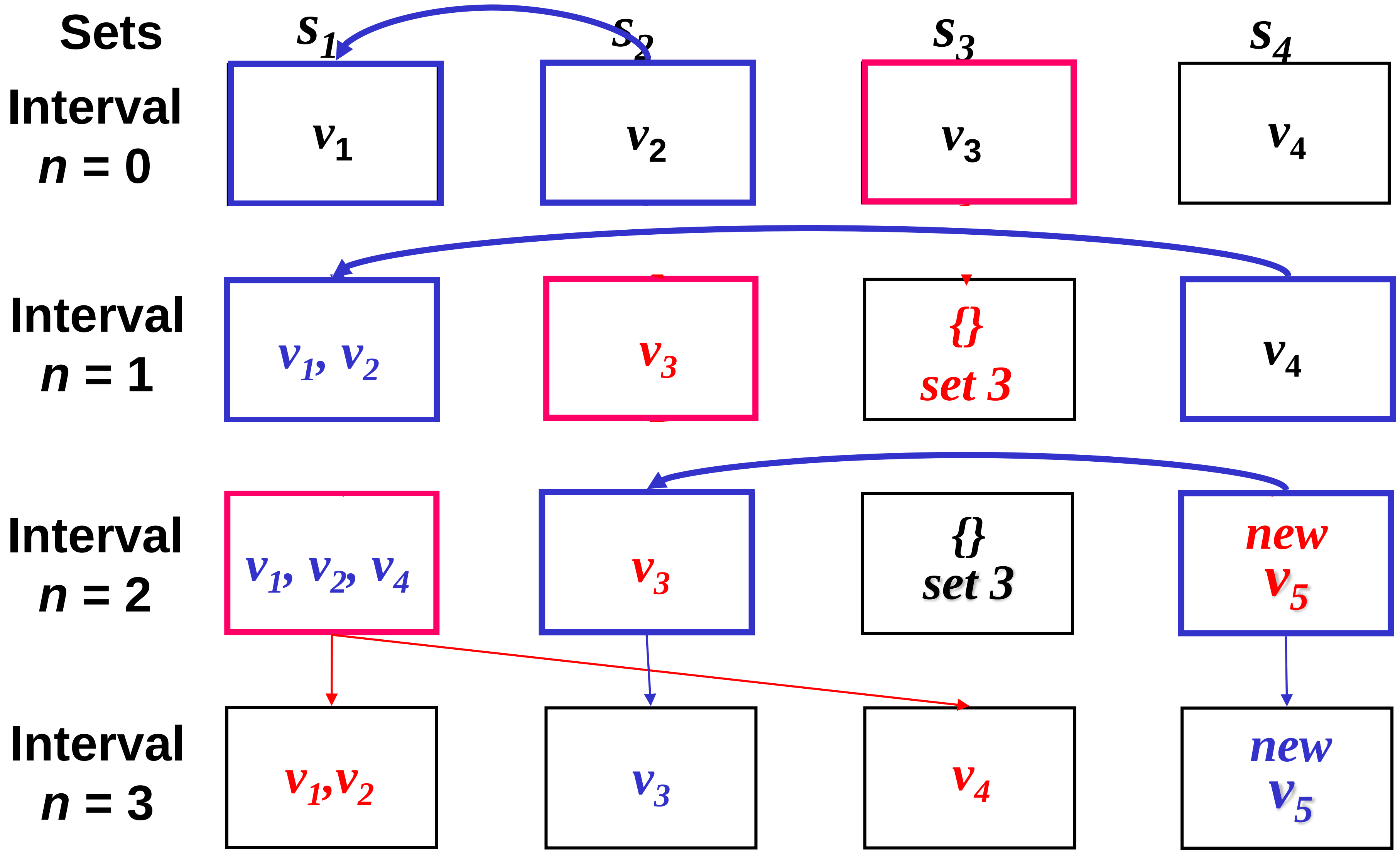
Discrete time interval: n

Child Word Probability: A singleton parent node probability will be halved by adding a new child (word). Similarly, ***FBL*** of a singleton with satellites will be halved and the child word ***FBL*** will equal to that of its parent after the process of splitting.

Word Real Probabilities: Word (node) real probability is equal to the sum of all the ***real*** probabilities of the given node and its decendent; Parent word and its decendent are in **Δ**.

Word State Probabilities: Word (node) state probability is equal to the sum of all the ***outset*** probabilities of the given node and its decendent; Parent word and its decendent are in ***D***.

S&M algorithm: the dictionary case, Δ=1.



Sets	s_1	s_2	s_3	s_4
Interval $n = 0$	v_1	v_2	v_3	v_4
Interval $n = 1$	v_1, v_2	v_3	{} set 3	v_4
Interval $n = 2$	v_1, v_2, v_4	v_3	{} set 3	new v_5
Interval $n = 3$	v_1, v_2	v_3	v_4	new v_5

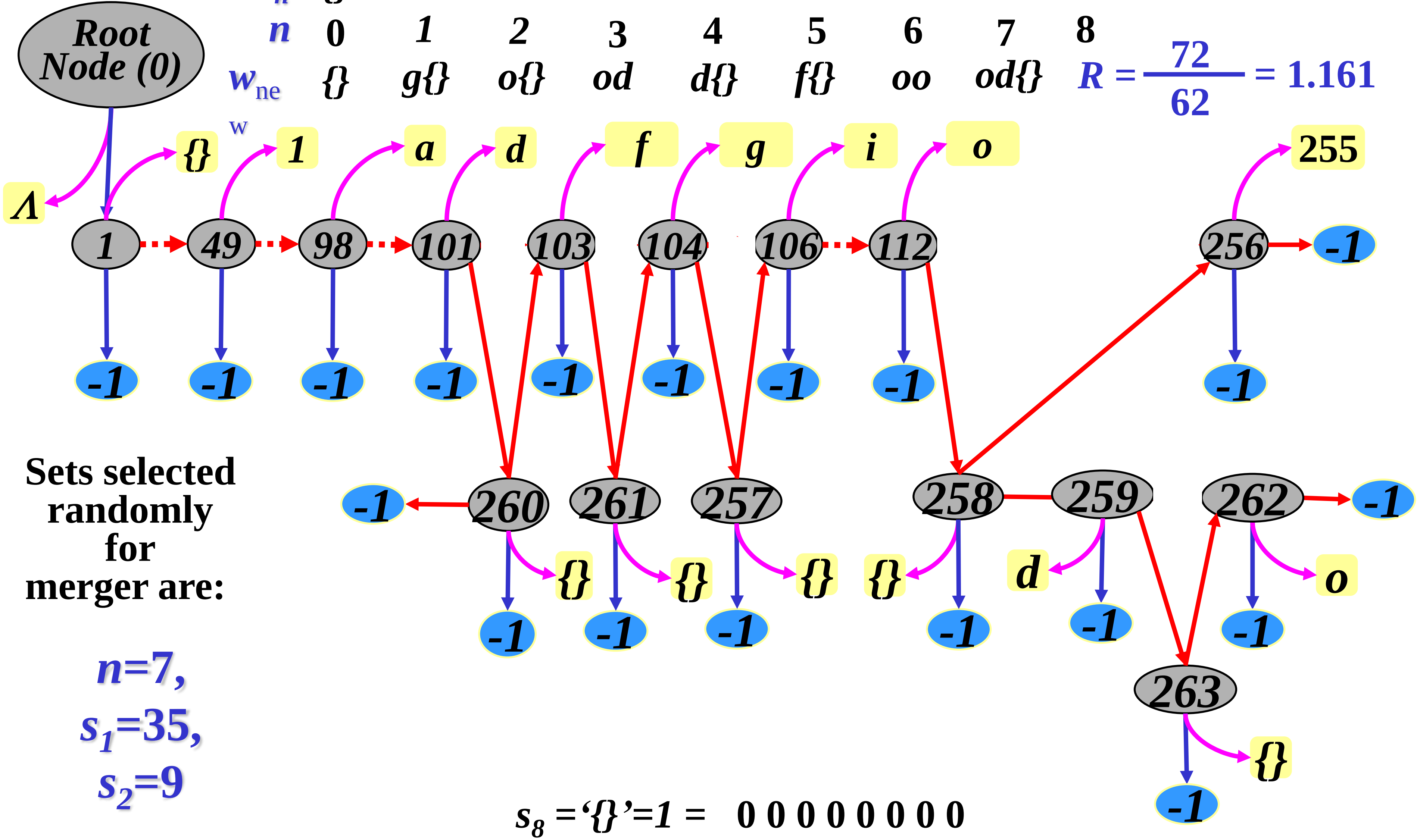
For large values of *n*, set probabilities will converge to values in the range of $2/N \ \delta \ p(s_i) \leq 1/N^{1/2}$

S&M algorithm: the dictionary case, Δ=1.

input string <goodfood{}>

s_n	{}	g~~o~~	o~~o~~	o~~d~~	d~~f~~	f~~o~~	o~~o~~	od~~{}~~	EOF
n	0	1	2	3	4	5	6	7	8
w_{ne}	{}	g{}	o{}	od	d{}	f{}	oo	od{}	

$$R = \frac{72}{62} = 1.161$$



Sets selected randomly for merger are:

$n=7$,

$s_1=35$,

$s_2=9$

s_8 =‘{}’=1 = 0 0 0 0 0 0 0 0

$L(w_{word8})$ is 8 bits

S&M algorithm: the dictionary case, Δ=1.

Results of Practical Simulation

Prob(Set1)

$$2/N \leq p(s_i) \leq 1/N^{1/2}$$

$$0.0078 \leq p(s_i)\ \delta\ 0.067$$

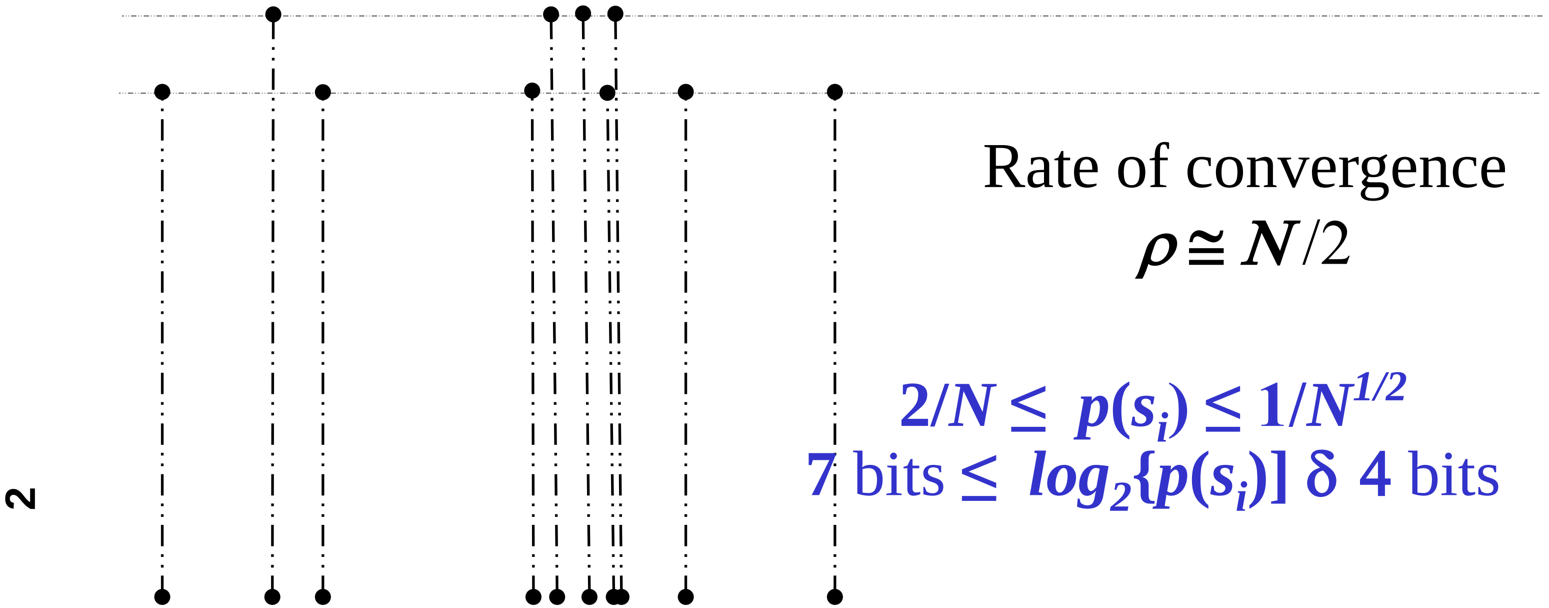
Rate of convergence

$$\rho \cong N/2$$

Abdullah Hashim-slide No. 8

S&M algorithm: the dictionary case, Δ=1.

$_2$Results of Practical Simulation



Rate of convergence

$$\rho \cong N/2$$

$$2/N \le p(s_i) \le 1/N^{1/2}$$

7 bits ≤ $log_2\{p(s_i)]$ δ 4 bits

3 db points ●-·-·–·-·–·-·–·-·–●

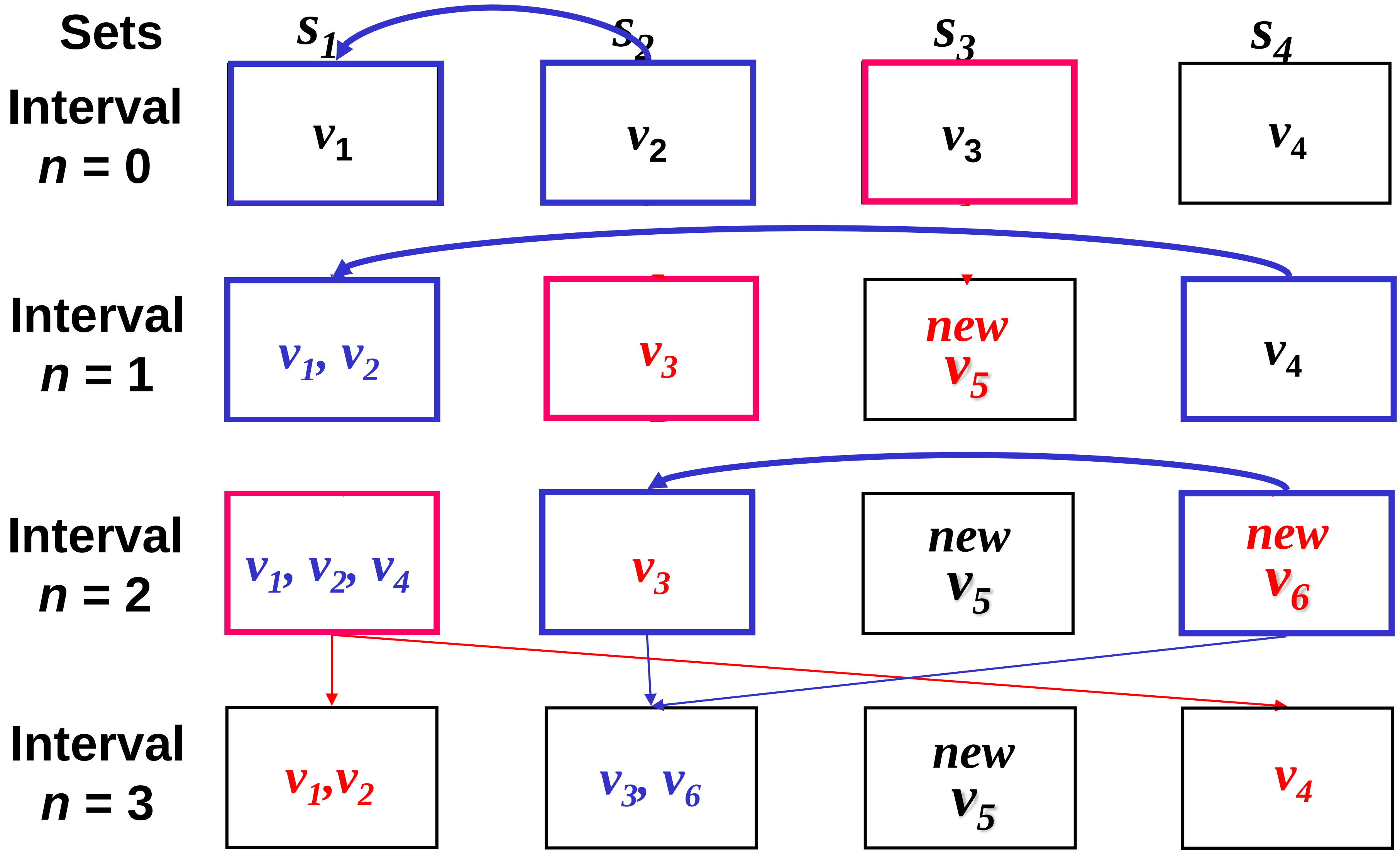
Abdullah Hashim-slide No. 9

S&M algorithm: the dictionary case

Note:

The proposed ***S&M-Dictionary case*** differs fundamentally from that of ***Lempel and Ziv*** scheme in that a new word will not be added to the extended dictionary (at the ***n***-th interval) unless the set is a singleton with Δ satellite sets. This condition ensures that only words of high frequency of occurrence are added to the dictionary and in turn increases the inclusion of longer words of high frequency at subsequent intervals. It is common practice that one of the major conditions of the dictionary pruning process is that the pruned word is of low wordlength and frequency of occurrence.

S&M algorithm: the dictionary case



Sets	s_1	s_2	s_3	s_4
Interval $n = 0$	v_1	v_2	v_3	v_4
Interval $n = 1$	v_1, v_2	v_3	new v_5	v_4
Interval $n = 2$	v_1, v_2, v_4	v_3	new v_5	new v_6
Interval $n = 3$	v_1, v_2	v_3, v_6	new v_5	v_4

For large values of *n*, set probabilities will converge to $2/|S|$

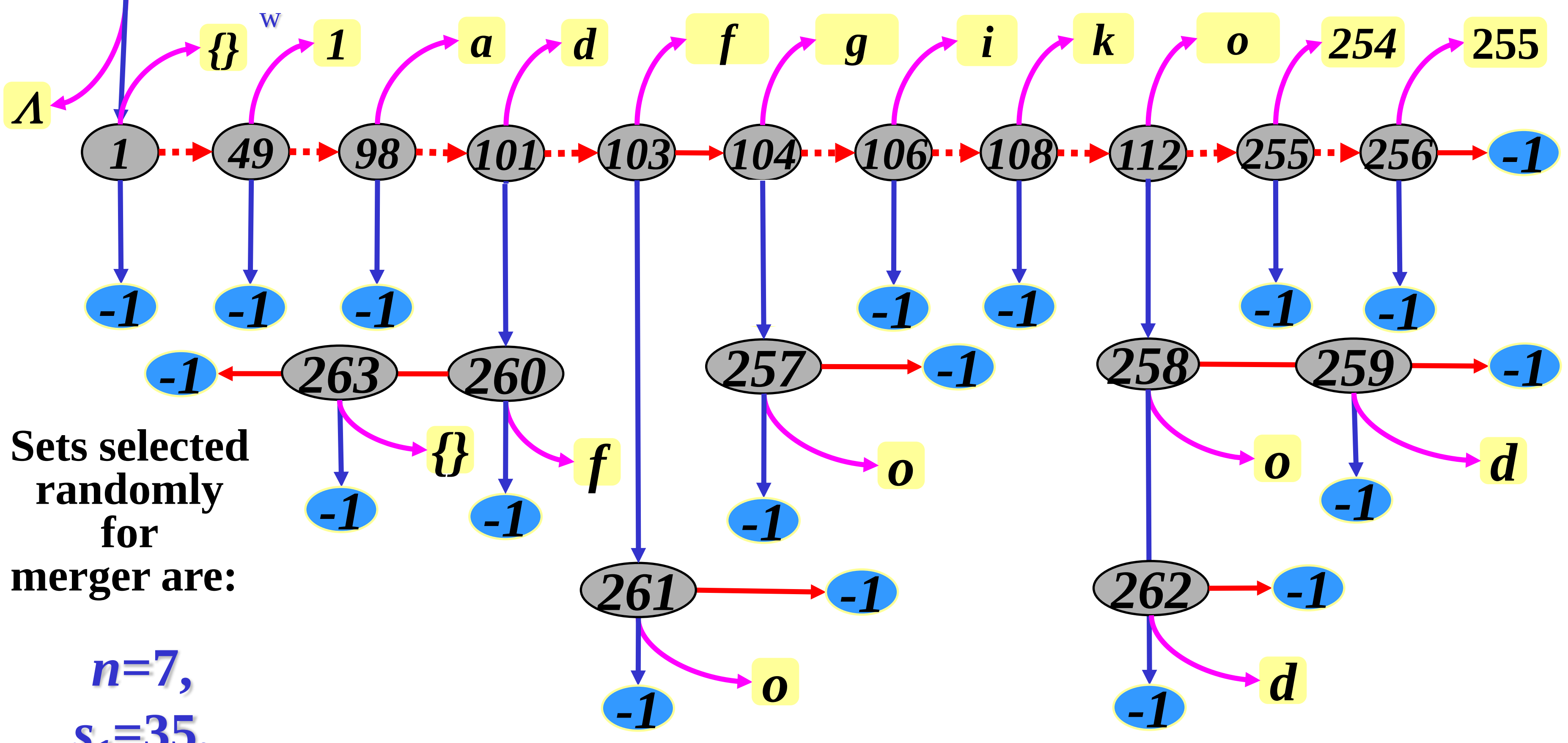
S&M algorithm: the dictionary case.

input string <goodfood{}>

s_n	{}	g~~o~~	o~~o~~	o~~d~~	d~~f~~	f~~o~~	oo~~d~~	d~~{}~~	EOF
n	0	1	2	3	4	5	6	7	8
w_{ne}	{}	go	oo	od	df	fo	ood	d{}	

w

$$R = \frac{72}{64} = 1.125$$



Sets selected randomly for merger are:

$n=7$,

$s_1=35$,

$s_2=9$

s_8 = '{}'=1 = 0 0 0 0 0 0 0 0

$L(w_{word8})$ is 8 bits

S&M algorithm: the dictionary case, Δ=0.

results of practical simulation

Prob(Set1)

$$\mathbf{2}/N \le p(s_i) \le \mathbf{1}/N^{1/2}$$

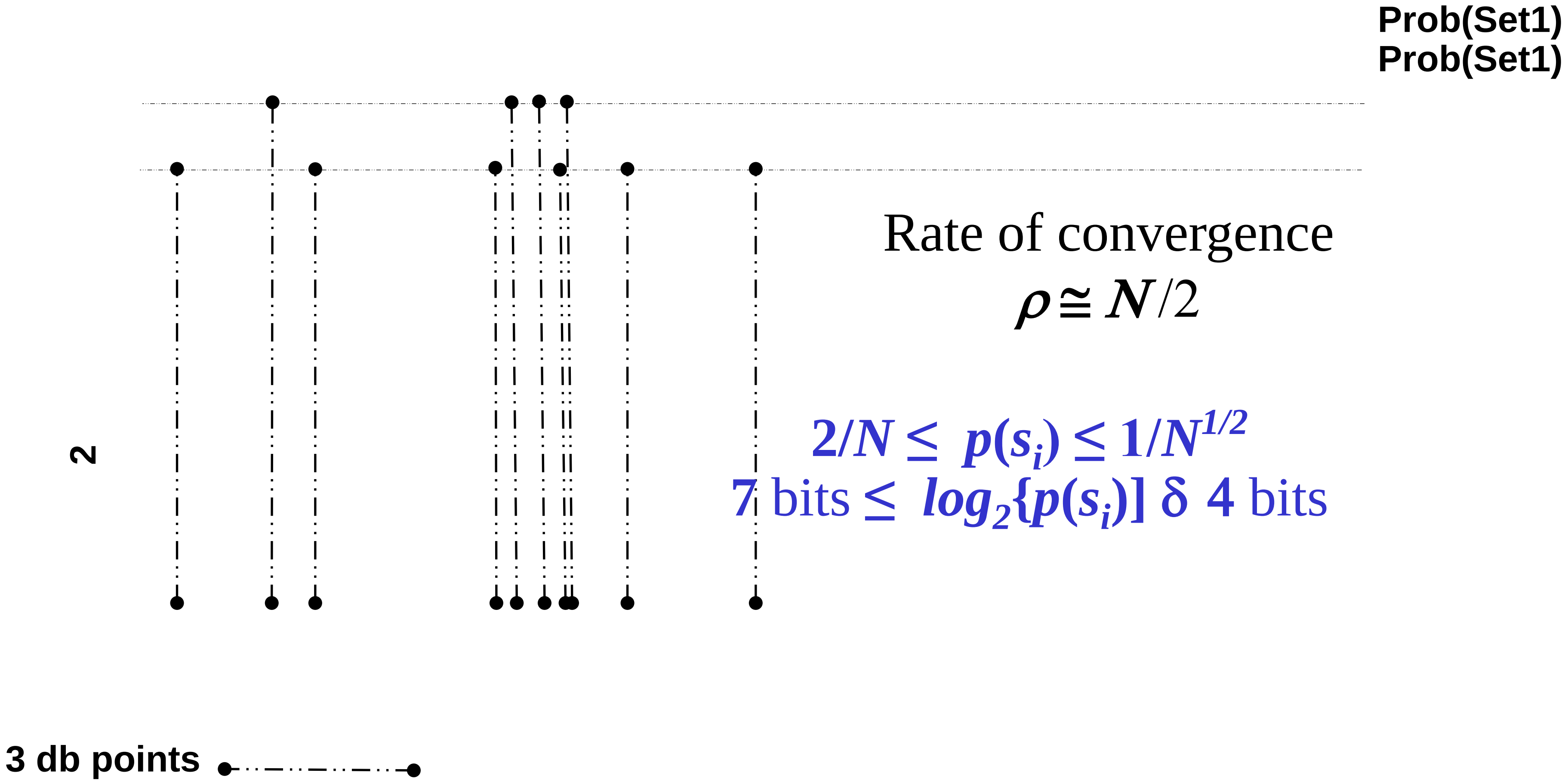
$$\mathbf{0.0078} \le p(s_i) \le \mathbf{0.067}$$

Rate of convergence

$$\rho \cong N/2$$

S&M algorithm: the dictionary case, Δ=0.

2Results of Practical Simulation



Rate of convergence

$$\rho \cong N/2$$

$$2/N \leq p(s_i) \leq 1/N^{1/2}$$

$$7 \text{ bits} \leq \log_2\{p(s_i)] \; \delta \; 4 \text{ bits}$$