A *Personal* View of Generalized Huffman Encoding

Mordecai J. Golin

Department of Computer Science
Hong Kong University
of Science and Technology

This talk presents an idosyncratic personal view of generalized Huffman Encoding developed through some of problems I've worked on. More specifically it will discuss:

- Unequal-Letter-Cost Huffman Codes
  - Non-Uniform Probability (the Karp problem)
  - Uniform probability (Varn Codes)
- Regular Language Huffman Codes
- Infinite Source Huffman Codes (Golomb Codes)

It will not discuss many other important generalizations, e.g.,

- Dynamic Huffman Codes
- Length-Limited Huffman Codes
- Alphabetic Huffman Codes (Hu-Tucker)
- Bidirectional Huffman Codes
- ⋮

For a good survey on generalizations on Huffman Encoding
"Code and Parse Trees for Lossless Source Encoding"
by Julia Abrahams
in *Sequences 1997*

Some History

In 1991 Dennis Shasha (who writes puzzle books) asked:

How would you solve the Huffman Encoding problem for Morse Code if "." (dots) and "-" (dashes) have different lengths

because he wanted it for a puzzle.

It sounded like an easy problem but wasn't.

We later found that it first appeared in the literature at least 40 years ago (Karp, 1961) but it's still unknown whether it's polynomial time solvable, NP-Hard, ...

What is Huffman Coding?

What is known

- **Karp 1961:**
  If α/β is rational, problem can be written as an Integer Linear Program (and solved in exponential time). This can be extended to r-ary alphabets with letters of unequal length $c_1 \leq c_2 \leq \ldots \leq c_r$.

- **Gilbert 1995**
  Heuristic Algorithms that work well in practice

- **Golin & Rote 1998**
  $O\left(n^{c_r+2}\right)$ dynamic programming algorithm
  We'll see similar algorithm later.

- **Bradford, Golin, Larmore, Rytter 1998**
  $O\left(n^{\beta}\right)$ for binary case when $\alpha \leq \beta$ are integers.
  Dynamic programming with Monge Property
  (Monotone Matrices)

BIG OPEN QUESTION:
It is still not known whether this problem is in P, NP-Hard, or ....

SMALL OPEN QUESTION:
How to solve when α/β is irrational, e.g., $\alpha = 1$, $\beta = \pi$.

Uniform Probability Case

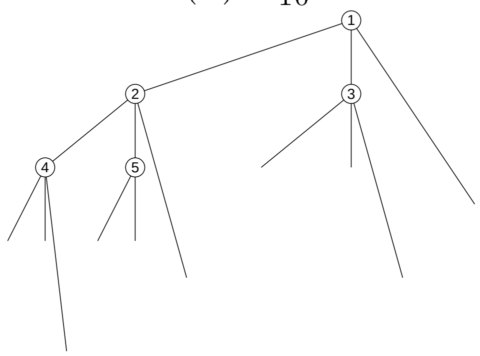
In this case we are given r letters with lengths

$$c_1 \leq c_2 \leq \ldots \leq c_r,$$

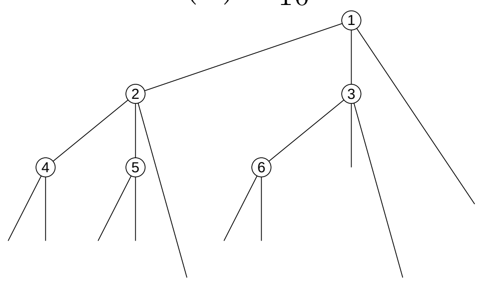
but uniform probabilities, i.e., $p_1 = p_2 = \ldots = p_n = \frac{1}{n}$.
To simplify problem we look at external path length of trees and try to find tree with minimum EPL.

Example: $(c_1, c_2, c_3) = (2, 2, 5)$ and $n = 10$.
The cost of the trees are, respectively, 60, 59, 60, 62.
(b) is an optimal tree for $n = 10$ leaves.

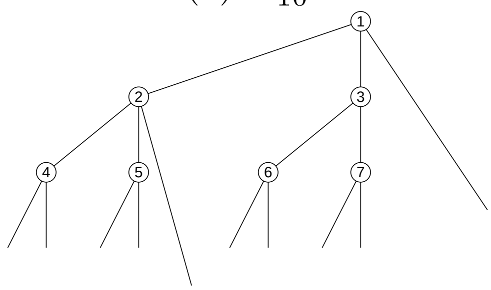
$(a)\ T_{10}^{5}$



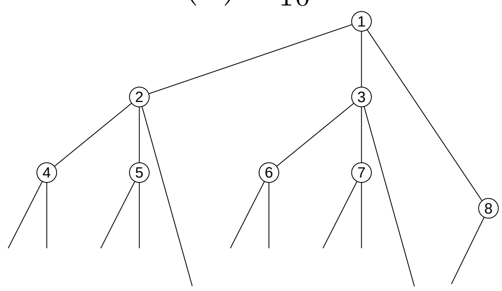
$(d)\ T_{10}^{6}$



$(c)\ T_{10}^{7}$



$(d)\ T_{10}^{8}$



Interesting for

- Coding (known as Varn Codes)
- Unequal Cost Searching
- Broadcast protocols (postal model of communication)

What's known for cost

Given $c_1 \leq c_2 \leq \ldots \leq c_r$, find the path length of the tree as a function of n, the number of leaves.

- **Csiszár: 1969**
- **Altenkamp and Mehlhorn: 1980**
- **Kapoor and Reingold: 1989**
  Complete analysis of binary case.
- **Savari: 1994**
  Some bounds in r-ary case.
- **Choi and Golin: 1996**
  Combinatorial analysis of r-ary case
  How tree changes as number of leaves grow.

Algorithms for building optimal trees

- **Varn 1971:**
- **Cot 1974:** $O(nr^2)$
- **Perl, Garey and Even 1975:** $O(nr^2)$ and $O(nr \log n)$
  Build optimal tree and fix
- **Choy and Wong 1983:**
  $O(n)$ algorithm when $c_i = \alpha + (i-1)\beta$ for fixed α, β.
- **Kapoor and Reingold: 1989**
  $O(n)$ Binary Case
- **Golin and Young: 1996** $O(n \log^2 r)$
  Constructs a small collection of trees guaranteed to contain optimal.
- **Choi and Golin: 1996** $O(n \log r)$
  Different than the others since it doesn't try to construct best tree. Instead it uses knowledge about how best tree changes as n grows and grows it.

Regular-Language Huffman Coding

In the literature one finds various types of constrained Huffman Codes. In these problems we are given $p_1 \geq p_2 \geq \ldots \geq p_n$ and are asked to find a minimal cost prefix-free code such that every word in the code belongs to some specified language L. As examples

- $L = (0+1)^*1$.
  One Ended Codes. Useful for synchronization
  Berger and Yeung: 1990
  Capocelli, De Santis and Persiano: 1994
- $L = \{w \in \{0,1\}^* \,:\, w \text{ contains at most } U \text{ 1's}\}$.
  "The Sound of Silence".
  Dolev, Korach, Yukelson. INFOCOM 1999
- L is the set of all words in $\{0,1\}^*$ such that between every two 1's there are between l and r 0s.
  (l,r) **Run-Length-Limited codes.**
- $L = \left(0^\alpha + 1^\beta\right)^*$
  Varn Languages.
  This is exactly the unequal-letter cost problem!

Notice that *all* of these languages are regular which is why we call them *Regular-Language Huffman Coding* problems.

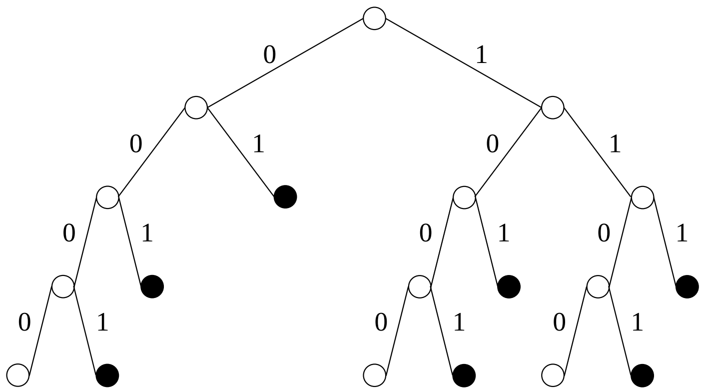
Meta-Theorem: Let L be a regular language accepted by a Deterministic Finite Automaton (DFA) with K states. Let $p_1 \geq p_2 \geq \ldots \geq p_n$.

Then a minimal-cost prefix-free code such that all codewords are in L can be constructed using dynamic programming in $O\left(n^{O(K)}\right)$ time.

The $O(K)$ term is always $\leq 2K+2$ but special DFA structures can reduce it even more.

In the next few slides we will see how this works for

$$L = (0+1)^*1.$$



Correspondence between One-Ended Prefix-Free Codes and trees: ‘0’ is a left edge and a ‘1’ a right edge.
Codewords correspond only to *right leaves*.

This tree is the code $\{0001,\ 001,\ 01,\ 1001,\ 101,\ 1101,\ 111\}$.

For T a tree with n right leaves, $P = \{p_1,\ p_2,\ p_3,\ \ldots,\ p_n\}$

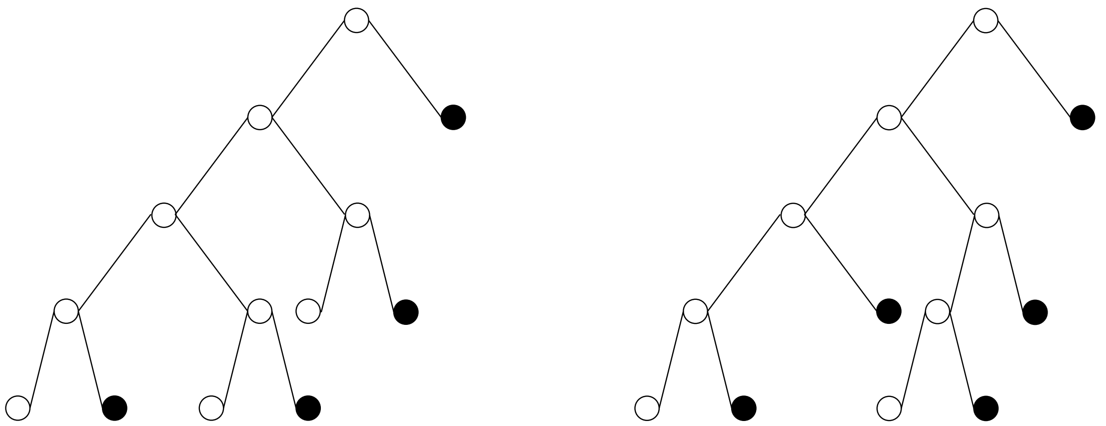
$$\mathrm{Cost}(T, P) = \sum_i \mathrm{depth}(v_i) \cdot p_i.$$

Given P, the optimal *One-Ended* code problem is equivalent to finding T with n right leaves that minimizes $\mathrm{Cost}(T, P)$.

Feasible Trees

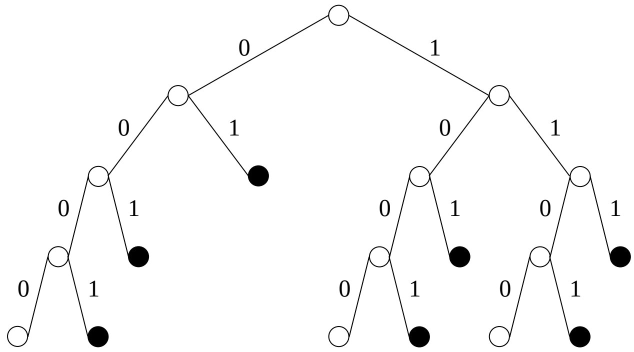
T is *full* if every internal node in T has two children.

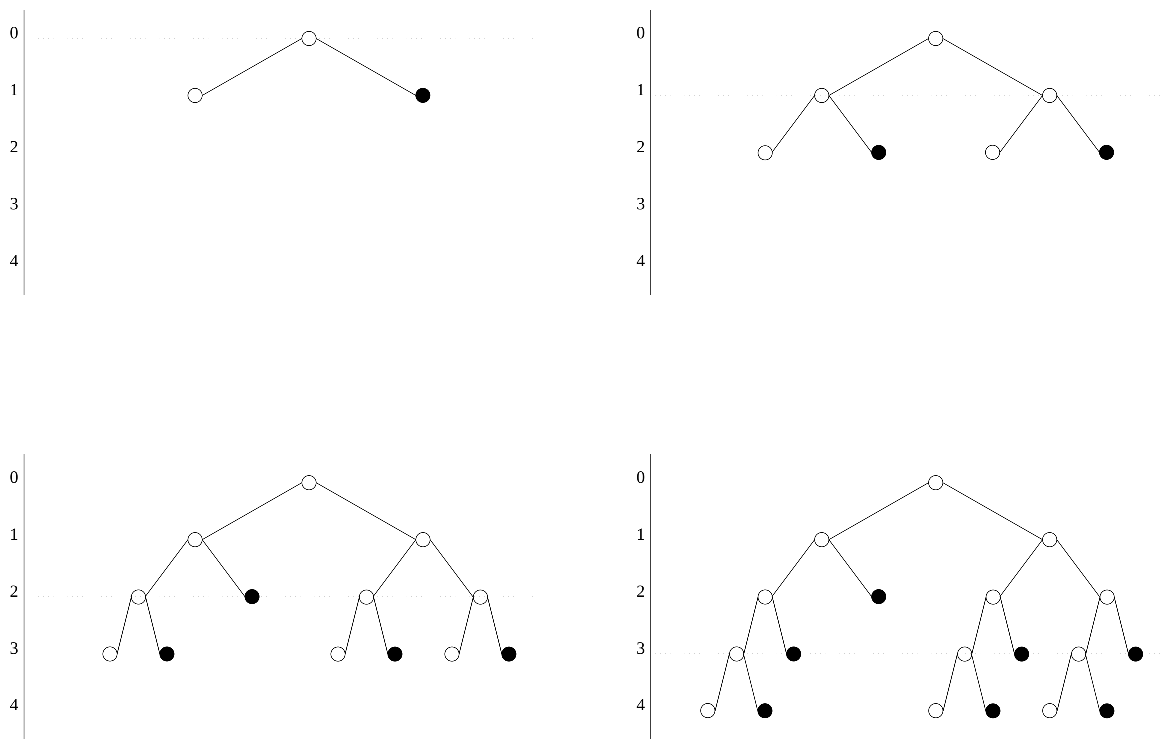
T is *feasible* if T is full and also satisfies
$\forall u \in T$ is a right node and internal then
    $\forall v \in T$ with $depth(v) = depth(u)$,
    and v a left node
     v is internal.



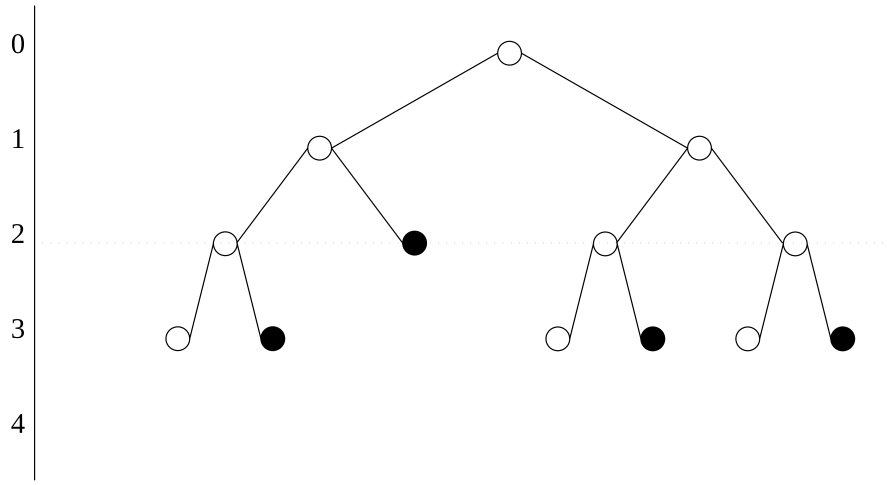
Lemma: Given P, there exists an optimal tree T that is feasible.

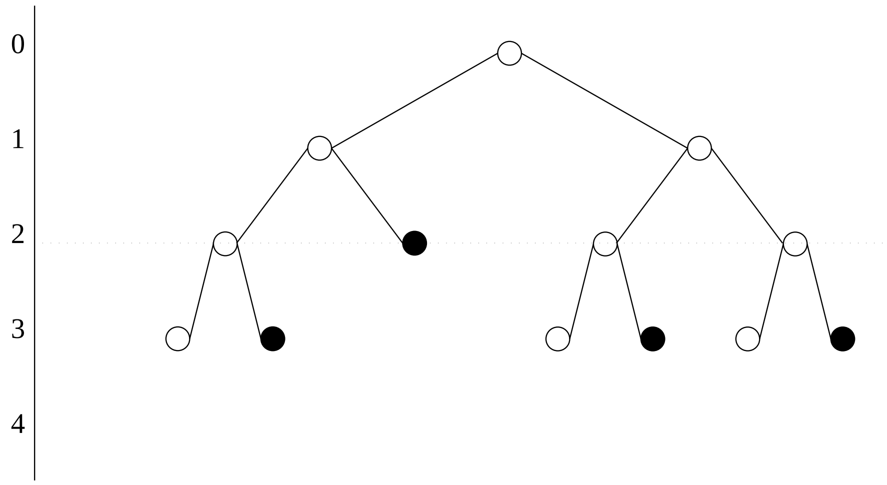
$\mathrm{Trunc}_i(T)$ contains root and all nodes of T at depth $\leq i+1$.





- Want to construct optimal trees top down.
- Construct "optimal" min-cost truncated trees top down by *expanding* smaller optimal trees into larger ones.
- Need to define "cost" of a truncated tree.
- Need to define "expanding"





Let T be a tree of depth at most $i+1$.
Set m to be the number of right leaves in T with depth $\leq i$.
Set b to be the number of right leaves in T at depth $i+1$.
Define $sig_i(T) = (m, b)$.

Now set $v_1, \ldots, v_m$ to be the right leaves with depth $\leq i$. ordered from top to bottom.

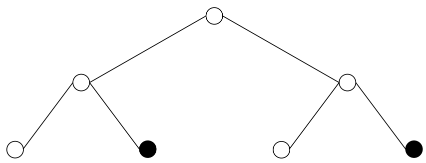
The i-level *partial cost* of T is defined to be

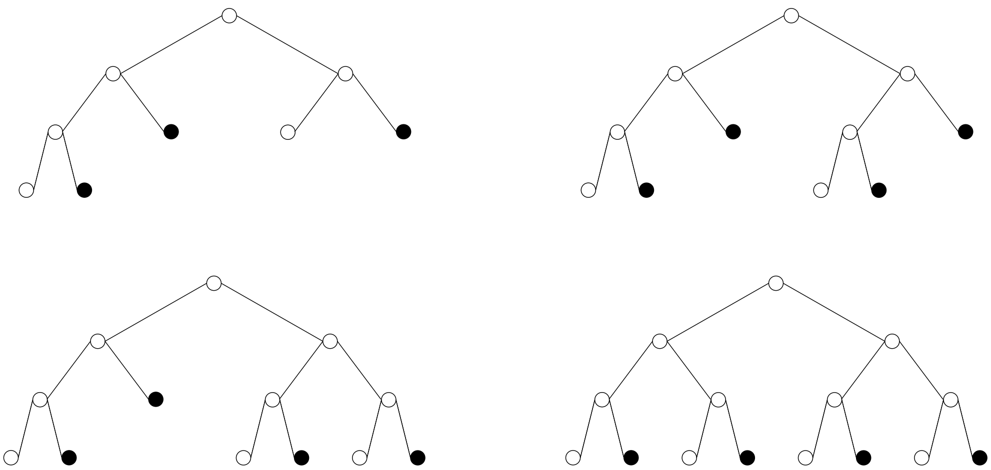
$$\mathrm{Cost}_i(T) = \sum_{t=1}^{m} \mathrm{depth}(v_t) \cdot p_t + i \cdot \sum_{t=m+1}^{n} p_t$$

Let T have $sig_i(T) = (m, b)$.

T can be *expanded* one level by making q of the $2b$ leaves at level $i+1$ into internal nodes.

To maintain feasibility use following rule:
  if $q \leq b$, make q left nodes at level $i+1$ internal
  if $q > b$, make all b left nodes and $q - b$ right nodes at level $i+1$ internal.



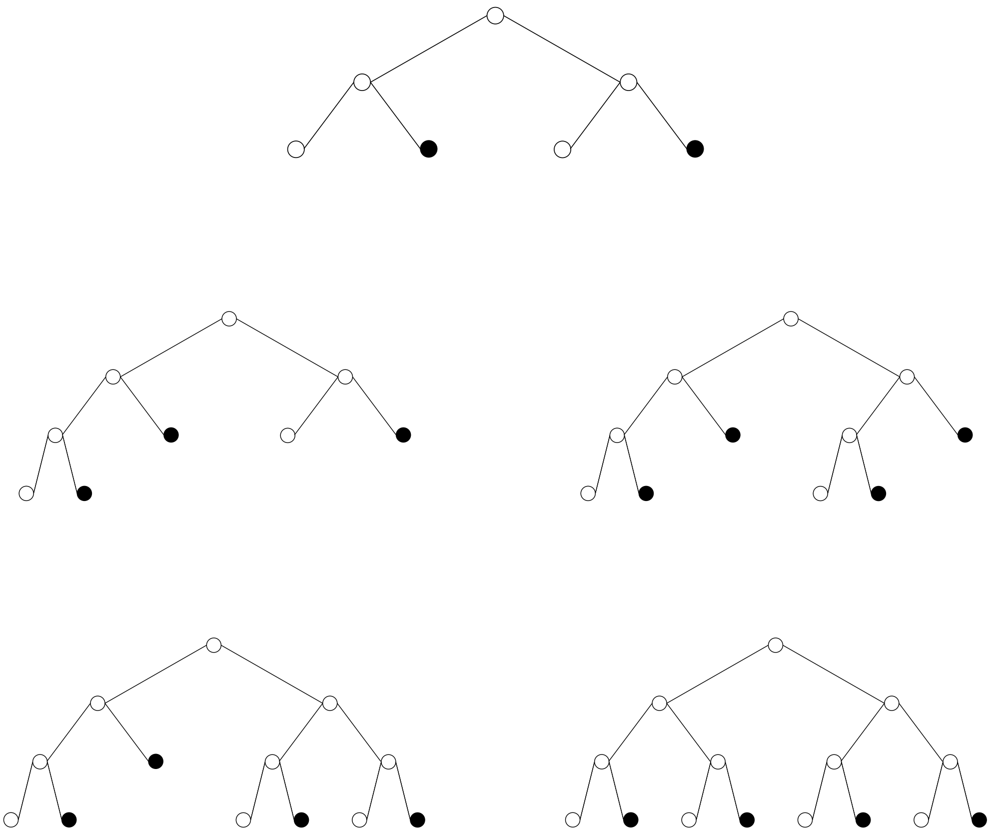


Lemma

Let T have $sig_i(T) = (m, b)$
and $T' = Expand(T, q) \quad (q \leq 2b)$

$$\mathrm{Cost}_{i+1}(T') = \mathrm{Cost}_i(T) + \sum_{m<t\leq n} p_t$$

- If $0 \leq q \leq b, \quad sig_{i+1}(T') = (m + b, q)$
- If $b + 1 \leq q \leq 2b, \quad sig_{i+1}(T') = (m + 2b - q, q)$



Idea: Instead of optimizing over trees (exponential number) define $OPT[m, b]$ to be minimum cost of a tree whose signature is (m, b) or formally:

$$\begin{aligned} OPT[m,b] &= \min\{\mathrm{Cost}_i(T) : \exists\, i, T, \\ &\qquad T \text{ is a feasible with } sig_i(T) = (m,b)\} \end{aligned}$$

and then optimize over all signatures ($O(n^2)$).

We use dynamic programming to fill in the $OPT[m, b]$ table

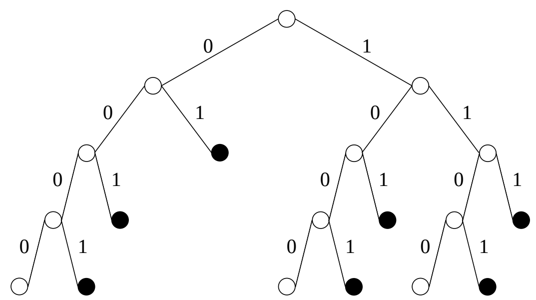
If T feasible with n leaves and depth i then

$$sig_i(T) = (n, 0)$$

So if if T' is an optimal feasible tree (with n right leaves) then

$$OPT[n, 0] = \mathrm{Cost}_i(T') = \sum_{t=1}^{n} \mathrm{depth}(v_t) \cdot p_t.$$

which is the real cost we want to minimize over.



Algorithm:
Fill in the $OPT[m, b]$ table
$OPT[n, 0]$ is minimum cost.
Backtracking yields tree with this cost.

Lemma: Set

$$\begin{aligned}\mathcal{M}(m',b') \;=\; \{(m,b)\,:\,&(m,b)\neq(m',b') \text{ and}\\ &\exists q \text{ s.t. } 0\le q\le b \text{ and } (m',b')=(m+b,q)\\ &\text{or } \exists q \text{ s.t. } b+1\le q\le 2b \text{ and } (m',b')=(m+2b-q,q)\}\end{aligned}$$

Then

$$OPT[m',b'] = \min\left\{OPT[m,b] + \sum_{m<t\le n} p_t \;:\; (m,b)\in\mathcal{M}(m',b')\right\}$$

with initial condition $\qquad OPT[0,1]=0$

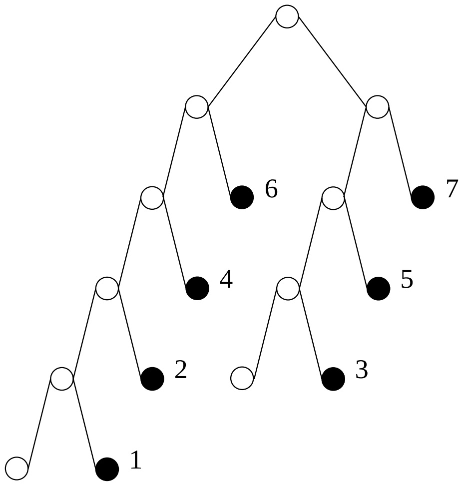
Dynamic programming recurrences!!

Can be used to fill in $OPT[m,b]$ table

Since table has only $O(n^2)$ entries and minimization recurrence uses $O(n)$ time, entire algorithm uses $O(n^3)$ time.
Complication is in finding proper order to fill in table.

Algorithm is from **Chan and Golin: 2000**.
Can be improved to $O(n^2)$.

	$b=0$	1	2	3	4	5	6	7
$m=0$	∞	$\mathbf{0}$	$\mathbf{28}$ $(\mathbf{0},\mathbf{1})$	∞	56 $(0,2)$	∞	∞	∞
1	28 $(0,1)$	28 $(0,1)$	49 $(1,1)$	56 $(0,2)$	70 $(1,2)$	∞	77 $(1,3)$	
2	49 $(1,1)$	49 $(1,1)$	$\mathbf{56}$ $(\mathbf{0},\mathbf{2})$	70 $(1,2)$	71 $(2,2)$	77 $(1,3)$		
3	64 $(2,1)$	64 $(2,1)$	70 $(1,2)$	71 $(2,2)$	77 $(1,3)$			
4	71 $(2,2)$	71 $(2,2)$	$\mathbf{71}$ $(\mathbf{2},\mathbf{2})$	77 $(1,3)$				
5	77 $(4,1)$	77 $(4,1)$	80 $(3,2)$					
6	77 $(4,2)$	$\mathbf{77}$ $(\mathbf{4},\mathbf{2})$						
7	$\mathbf{78}$ $(\mathbf{6},\mathbf{1})$							



A generalization of this algorithm can be used to solve the **Regular-Language Huffman encoding problem** in $O(n^{O(K)})$ when the codewords are restricted to belong to some regular language L accepted by a DFA with K states.

Restricted to the Varn Languages this is exactly the dynamic programming algorithm for the unequal-cost Huffman problem.

The idea is that we again build the trees type down; the signature will keep track of how many nodes "in" each state there are at that level.

OPEN QUESTIONS:

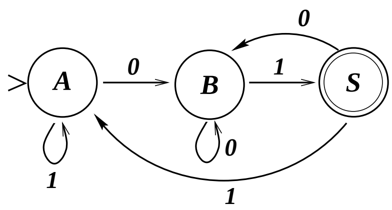
- Is there a better algorithm?
- Is the Regular-Language Huffman encoding problem NP-Hard?

The Uniform Probability Case

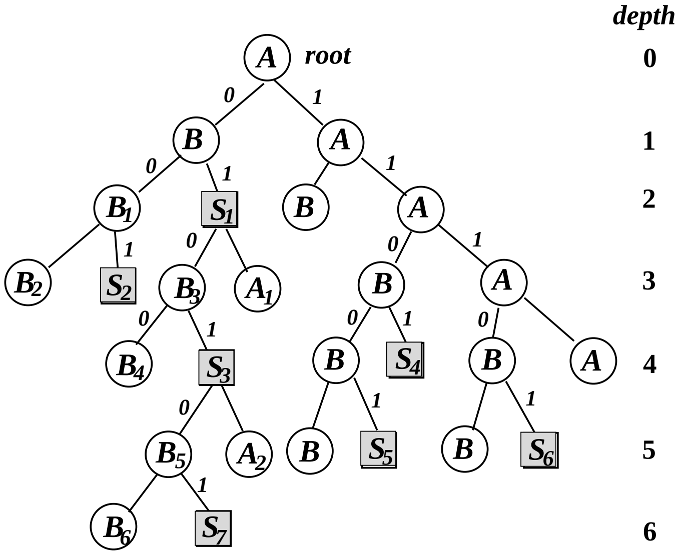
We still assume that all of the codewords must belong to some given regular language L but now also assume that all of the leaf probabilities are equal.

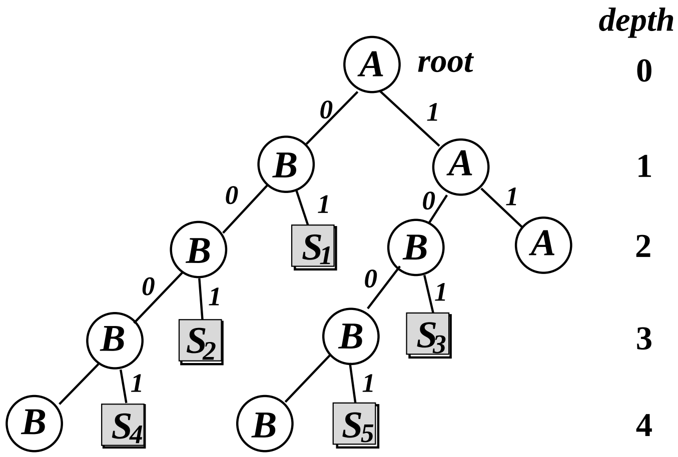
The problem is to find a minumum cost tree with n good leaves in which a "good leaf" is in L and the cost of the tree is the path length of the good leaves.

Example: suppose L is the set of all strings ending in 01.



and we want to build a min-cost tree with $n = 5$ good leaves. (top tree has cost = 23; bottom has cost=16 and is optimal).





We'd like a "translator" that can take a DFA accepting L and automatically derive the combinatorics of the optimal trees as n, the number of good leaves, grows.

Knowing the combinatorics will imply both good algorithms for constructing such trees and derivation of the path length as a function of the number of leaves.

Golin and Na: 2001 does this for DFAs with only one accepting state (Varn Languages have only one accepting state)

For general DFAs, "result" is not known.

For any particular DFA it does not seem too difficult.
The problem is coming up with a general purpose translator.

Infinite Source Coding

Huffman coding usually assumes a *finite* number of probabilities (weights)

$$p_0 \geq p_1 \geq p_2 \geq \cdots \geq p_{n-1}$$

What happens if there are *countably infinite* p_is?

$$p_0 \geq p_1 \geq p_2 \geq \cdots$$

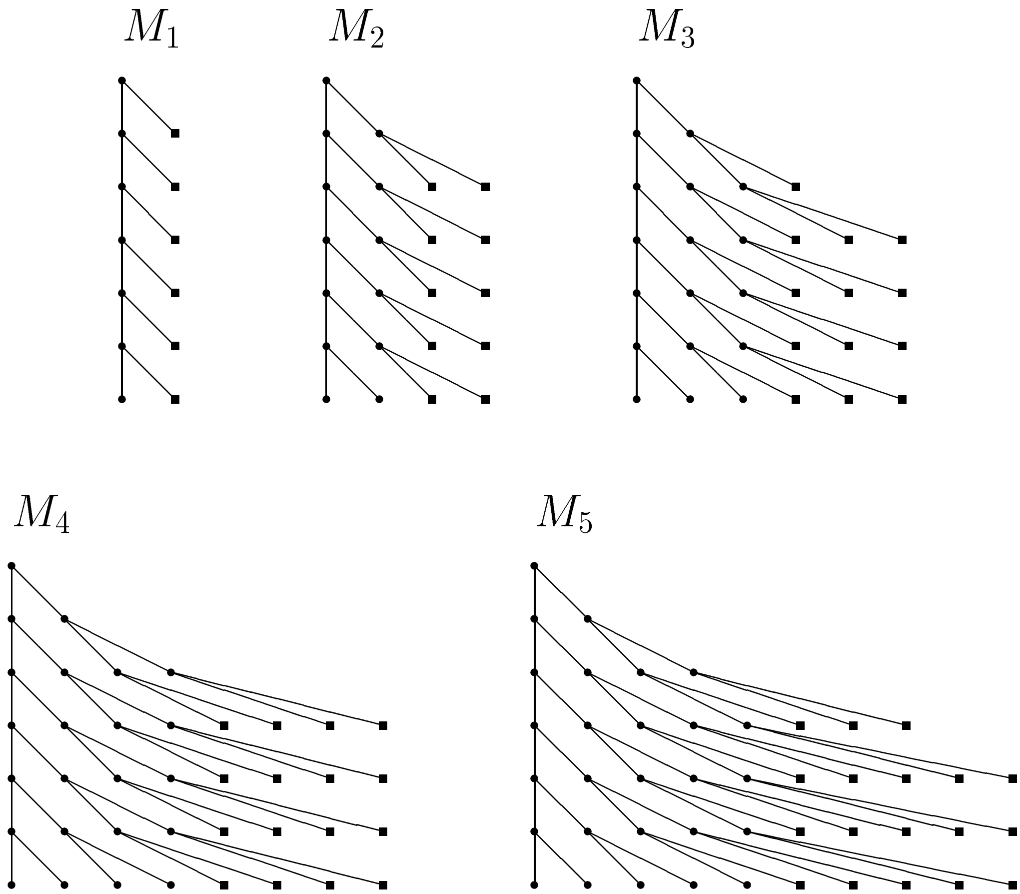
This first arose in Run-Length Encoding of strings produced by an infinite geometric source (Golomb 1966):

- string of **A**s and **B**s
- each character generated independently.
  Probability of **B** is p; of **A**, $1-p$.
- For $i = 0, 1, 2, \ldots$ set $X_i = \underbrace{BB\ldots BB}_{i \text{ times}} A$.
- An infinite string can be uniquely witten as concatenation of X_is
- probability of X_i occurring is $(1-p)p^i$.
- So we want to construct a Huffman code for infinite source with $p_i = (1-p)p^i$.

Theorem 0.1 *(Gallager and Van Voorhis (1975)) Given p, let m be the unique integer that satisfies*

$$p^m + p^{m+1} \leq 1 < p^m + p^{m-1}.$$

Then the trees below are optimal for the geometric distribution with $p_i = (1-p)p^i$.



Some Other Work

- **Abrahams 1994:**
  Extended results to ternary alphabets
- **Kato, Han, Nagaoka 1996:** generalized problem to case where infinite source has infinite entropy
- **Linder, Tarokh and Zeger 1997:** proved that if an infinite source's entropy is finite then Optimal Infinite Tree always exists.
- **Merhav, Serrousi amd Weinberger: (1997)** derived optimal trees for infinite *two-sided* geometric distributions.
- **Golin(1998)**: proved uniqueness of optimal trees

Open Problems

1. The only distributions for which results are known are geometric and variants.

   Open question is how to find optimal code for other infinite sources, e.g., Zeta-Source: $p_i = i^{-k}$ for fixed $k > 1$.

2. Even for the geometric distribution very little is known for the unequal-cost case or regular-language case.

   Given the constraints on the code and a *fixed* p there is an algorithm for finding the optimal infinite code when $p_i = p^i(1-p)$. For example, given $p = \frac{1}{5}$ we can find the optimal code with $\alpha = 1$, $\beta = 2$.

   But, there is no general theorem, a la Gallager and Van Voorhis, showing how the optimal $(1, 2)$-trees change as p changes.

Gallager and Van Voorhis proved their result by finding a countable sequence of *finite* distributions that converge to the infinite distribution, constructing the Huffman trees for each of the finite distributions and then showing that those trees "converge" to the infinite tree.

This proof strongly used the fact that there is a local criteria for testing optimality of trees. G & V showed that their finite trees were optimal for their finite distributions. In generalized Huffamn Encoding no such criteria seems to exist.

Conclusion

- Huffman encoding is a fun source of problems
- Any small change to Huffman encoding seems to make problem much harder.
- How hard is not known. NP-Completeness proofs are long overdue (if they exist)
- Pointers to new problems that fit in the regular-language framework would be appreciated

References

[1] Julia Abrahams, "Huffman-Type Codes for Infinite Source Distributions," *Journal of the Franklin Institute,* **331B**(3) (1994) 265-271.

[2] Julia Abrahams, "Code and Parse Trees for Lossless Source Encoding," *Sequences 1997,* (1997).

[3] Doris Altenkamp and Kurt Mehlhorn, "Codes: Unequal Probabilities, Unequal Letter Costs," *JACM,* **27**(3) (July 1980) 412-427.

[4] Toby Berger and Raymond W. Yeung, "Optimum "1"-Ended Binary Prefix Codes," *IEEE Transactions on Information Theory ,* **36**(6) (November, 1990) 1435-1441.

[5] Phil Bradford, Mordecai J. Golin, Lawrence L. Larmore, and Wojciech Rytter, "Optimal Prefix-Free Codes for Unequal Letter Costs and Dynamic Programming with the Monge Property," *Sixth European Symposium on Algorithms (ESA'98) (LNCS 1461),* (1998) 43-54.

[6] Capocelli, R.M., De Santis, A., and Persiano, G., "Binary prefix codes ending in a "1"," *IEEE Transactions on Information Theory ,* **40**(4) (July 1994) 1296-302.

[7] Chan Sze-Lok and M. Golin, "A Dynamic Programming Algorithm for Constructing Optimal "1"-ended Binary Prefix-Free Codes," *IEEE Transactions on Information Theory,* **46** (4) (July 2000) 1637-44.

[8] Siu-Ngan Choi and M. Golin, "Lopsided trees: Algorithms, Analyses and Applications," *Proceedings of the 23rd International Colloquium on Automata Languages and Programming (ICALP '96),* (July 1996) 538-549.

[9] D.M. Choy and C.K. Wong, "Construction of Optimal alphabeta Leaf Trees with Applications to Prefix Code and Information Retrieval," *SIAM J Comput,*, **12**(3) (Aug 1983) 426-446..

[10] N. Cot, "A Linear-Time Ordering Procedure With Applications to Variable Length Encoding," *Proccedings of the 8th Annual Princeton Conference on Information Sciences and Systems,* (1974) 460-463.

[11] I. Csiszár, *Simple Proofs of Some Theorems on Noiseless Channels,* (1969).

[12] Dolev, S., Korach, E., and Yukelson, D., ,
"The Sound of Silence: Guessing Games for Saving Energy in Mobile Environment," *Eighteenth Annual Joint Conference of IEEE Computer and Communications Societies (IEEE INFOCOM'99),* (1999) 768-775.

[13] E. N. Gilbert, "Coding with Digits of Unequal Costs," *IEEE Trans. Inform. Theory* , **41** (1995) 596–600 .

[14] Mordecai J. Golin, "A Combinatorial Approach to Golomb Forests," *Theoretical Computer Science (Special issue for work presented at the 1997 International workshop on Combinatorics and Computer Science),* **263**(1-2) (July 2001) 283-304.

[15] M. Golin and G. Rote, "A Dynamic Programming Algorithm for Constructing Optimal Prefix-Free Codes for Unequal Letter Costs," *IEEE Transactions on Information Theory,* **44**(5) (September 1998) 1770-1781.

[16] Mordecai Golin and Neal Young, "Prefix Codes: Equiprobable Words, Unequal Letter Costs," *SIAM Journal on Computing* , **25**(6) (December, 1996) 1281-1292.

[17] Sanjiv Kapoor and Edward M. Reingold, "Optimum Lopsided Binary Trees," *Journal of the Association for Computing Machinery,* **36** (3) (July 1989) 573-590.

[18] R. M. Karp, "Minimum-Redundancy Coding for the Discrete Noiseless Channel," *IRE Transactions on Information Theory,* **7** (1961) 27-39.

[19] A. Kato, T.S. Han, and H. Nagaoka, "Huffman Coding with Infinite Alphabet," *IEEE Transcations on Information Theory,* **42**(3) (May 1996).

[20] Tamas Linder, Vahid Tarokh, and Keneth Zeger, "Existence of Optimal Prefix Codes for Infinite Source Alphabets," *IEEE Transcations on Information Theory,* **43**(6) (1997) 2026 -2028.

[21] Neri Merhav, Gadiel Seroussi, and Marcello J. Weinberger, "Optimal Prefix Codes for Two-Sided Geometric Distributions (Abstract)," *Proceedings of the International Symposium on Information Theory,* (1997) 71.

[22] Y. Perl, M.R. Garey, and S. Even, "Efficient Generation of Optimal Prefix Code: Equiprobable Words Using Unequal Cost Letters," *Journal of the ACM,* **22**(2) (April 1975) 202-214.

[23] Serap A. Savari, "Some Notes on Varn Coding," *IEEE Transactions on Information Theory,* **40**(1) (January 1994).

[24] B.F. Varn, "Optimal variable Length Codes (Arbitrary Symbol Costs and Equal Code Word Probabilities)," *Informat. Contr., ,* **19** (1971) 289-301.