社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

最適アルファベット順符号を用いた コミュニケーションエイドの文字選択法

小山　智史†

† 弘前大学教育学部　〒036–8560 青森県弘前市文京町 1
E-mail: †koyama@hirosaki-u.ac.jp

あらまし　重度肢体不自由者用の入力法を設計する際に重要なことは，スイッチ操作に対応する候補文字群をわかりやすく表示することと，少ない操作回数で入力できるようにすることである．文字の出現確率がわかっている場合，ハフマン符号を用いると最小の平均操作回数で入力することができる．しかし，符号語の並びはアルファベット順にならず，文字盤表示は見やすくできない．本稿では，この問題がハフマン符号の代わりに最適アルファベット順符号を用いることにより解消できることを示す．次に，最適アルファベット順符号はハフマン符号よりも平均操作回数が増加（劣化）することから，この増加を抑える方法として「例外を許す最適アルファベット順符号」を提案する．それぞれの符号化入力法について，操作の様子と文字盤表示を具体的に示すとともに，平均操作回数をシミュレーションにより比較した．

キーワード　コミュニケーションエイド, 文字盤レイアウト, アルファベット順, ハフマン符号

A Selection Scheme for a Communication Aid Using the Optimal Alphabetic Code

Satoshi KOYAMA†

† Faculty of Education, Hirosaki University　1 Bunkyo cho, Hirosaki-shi, Aomori, 036–8560 Japan
E-mail: †koyama@hirosaki-u.ac.jp

Abstract　When we design a selection scheme of a communication aid for the severely disabled, it is important to provide a prompt that is fairly intuetive and to reduce switch activations as much as possible. If the probability of each letter is known, a Huffman code is optimal in the sense of minimal switch activations. But the code words cannot be ordered alphabetically. In this paper, this problem is solved by using the optimal alphabetic code instead of a Huffman code. The average code length of the optimal alphabetic code is larger than that of a Huffman code, so a newly modified code “optimal alphabetic code with exception” is proposed here. Selection schemes to determine an intended letter are expressed, and the simulation results of the average code length are shown.

Key words　Communication aid, Board layout, Alphabetical order, Huffman code

1. はじめに

1〜2 個のスイッチを用いて文字を選択する重度肢体不自由者用の入力法として，符号化入力法がある．行列スキャン方式も符号化入力法のひとつとみることができる [1], [2].

コミュニケーションエイドで重要なことは，スイッチ操作に対応する候補文字群がわかりやすく表示されることと，少ない操作回数で入力できることである．ここでは 2 個のスイッチを用いることとし，スイッチ操作で二者択一の候補文字群のいずれかを選択することを繰り返して目的の文字を決定する場合について考える．

符号化入力法では，利用者は入力したい文字の符号語を知っている必要がある．印刷した符号語表を手元に置いてもよいが，左と右のスイッチを用いるのであれば，候補文字群が画面の左右に分かれて表示されるとわかりやすい [1]（例えばハーティーラダー [3] の「2 ボタン 2 分割モード」）．行列スキャン方式の場合は候補文字群が画面上でハイライト表示され，利用者へのわかりやすいプロンプトとなっている．

符号化入力法では，文字の平均符号語長が 1 文字入力するための平均操作回数に相当する．文字の出現確率がわかっている場合，出現確率が高い文字が少ない操作回数で入力できるようにする工夫が，さまざまな形で行われてきた．例えば，文字盤

の左上に「_ E T A ...」などの出現確率の高い文字を配置する TIC [4] や，文字の出現確率に対して最適な符号であるハフマン符号 [5], [6] を使う方法 [7], [8] などがある．

文字の出現確率は文脈によって大きく変化する．n-gram にもとづく確率言語モデルとして PPM [9] がある．確率言語モデルを用いて 1 文字入力する毎に次文字の出現確率を計算し，それにもとづいて文字盤の配置や符号を最適にする方法が検討されている．行列スキャン方式では，1 文字入力する度に行列文字盤の配置が変わるのでは使いづらいため，固定の文字盤に文脈に応じた予測文字を追加配置する方法は一定の効果が認められた [10].

Roark は言語モデルとハフマン符号を組み合わせることにより，大きな効果が得られること示した [11]．しかしながら，ハフマン符号の符号語の並びはアルファベット順にならない．このため，スイッチ操作に対応する文字盤上の選択候補文字群は文字盤全体に散らばってしまう．ハイライト表示や符号語表示により操作に必要な情報は与えられるものの，分かりやすい表示とは言い難い．

この問題は，最適アルファベット順符号(注1)を用いることにより解消できる [6]．最適アルファベット順符号は Hu-Tucker アルゴリズム [12] や Garsia-Wachs アルゴリズム [13] で作ることができる．本稿では，最適アルファベット順符号を用いた入力法を具体的に示す．

最適アルファベット順符号はハフマン符号よりも平均操作回数が多少増加（劣化）する．この増加を抑える方法として，「例外を許す最適アルファベット順符号」を提案し，符号作成の手順を示す．

2 章で等長符号，行列スキャン方式，ハフマン符号，最適アルファベット順符号，例外を許す最適アルファベット順符号のそれぞれについて，操作の様子と文字盤表示を具体的に示し，3 章でシミュレーションによりそれぞれの符号化入力法の平均操作回数を比較する．

2. 符号化入力法

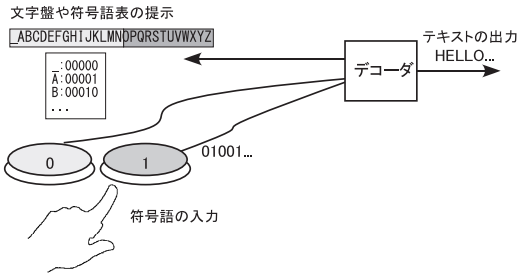
2.1 符号化入力法と行列スキャン方式

符号化入力法では，利用者は入力したい文字を符号語で入力する（図 1(a)）．ここでは 2 元符号を想定し，「0」と「1」に対応する 2 個のスイッチを用いることとし，アルファベットは $A = \{\texttt{\_},\texttt{A},\ldots,\texttt{Z}\}$ ($|A| = 27$) とする．

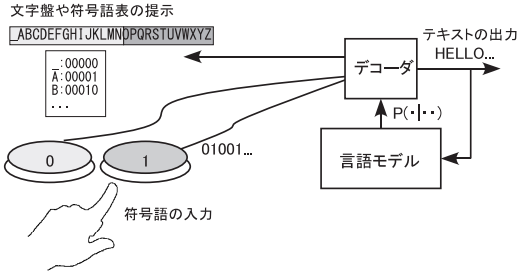
符号の例として「_（スペース）」を「00000」，「A」を「00001」のように，27 文字を 5 ビットで表わす等長符号がある．この場合の符号語長は 5 で，これがスイッチの操作回数である．この方法で「F」の符号語「00110」を入力する時の具体的な操作は図 2(a) のようになる．二者択一の候補文字群が図のようにディスプレイ上で左右に分かれて表示され，2 個のスイッチを左右に並べれば，対応関係はわかりやすい [1].

行列スキャン方式も符号化入力法の 1 種と考えることができる．図 2(b) は，他との比較のために文字盤の表示を 1 行で表したものである．はじめに行に相当するブロックを選択し，次にそのブロックの中から列に相当する文字を確定する．この例では「1（次の行）0（行を選択）0（列を選択）」の操作（符号語）で「F」が入力されている．

(注1)：本稿では，符号語の並びがアルファベット順である平均符号語長が最小となる符号を最適アルファベット順符号と呼ぶ．

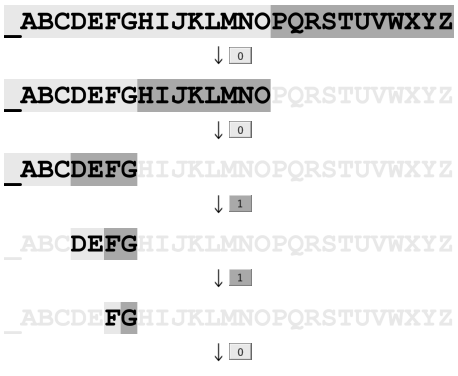


(a) 単純な符号化入力法

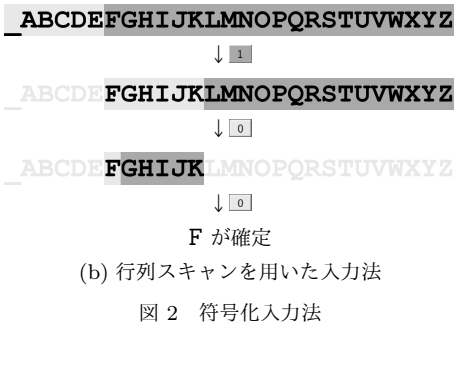


(b) 言語モデルを用いた符号化入力法

図 1　符号化入力法を用いた文字入力



(a) 等長符号を用いた入力法



(b) 行列スキャンを用いた入力法

図 2　符号化入力法

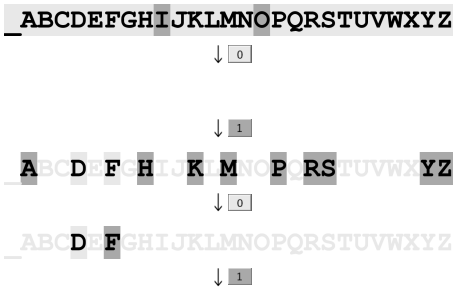
2.2 言語モデルを用いた符号化入力法

言語モデルを用いた符号化入力法は図 1(b) のように表すことができる．

n-gram による確率言語モデルを用いると，直前 $(n-1)$ 文字

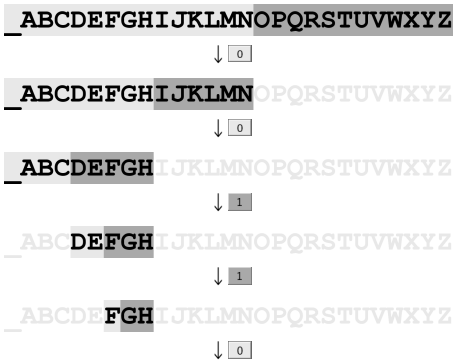
F が確定

(c) ハフマン符号を用いた入力法（符号語順表示）



F が確定

(d) ハフマン符号を用いた入力法（アルファベット順表示）



F が確定

(e) 最適アルファベット順符号を用いた入力法（**GW**）

F が確定

(f) 例外を許す最短アルファベット順符号を用いた入力法（**GW+**）

図 2 符号化入力法（続き）

の文脈 s に基づく次文字 a の確率 $P(a|s)$ が得られる．符号はこの確率に基づき作られる．

適応言語モデルの場合は，文字を入力する度に言語モデルを更新し，より良いモデルに変えていく．「学習」が進むにつれ，より精度の高い確率が推定できるようになる．

言語モデルを用いた符号化入力法は入力効率の面では優れているものの，文脈に応じて（つまり 1 文字入力する毎に）符号が変化するため，操作を促すための「わかりやすい表示」をどう工夫するかが重要である．

次節以降および対応する図 2(c)〜(f) は，言語モデルとして PPM（$o = 4$, Method C）[9] を用い，初期状態（言語モデルが空の状態）から

`_DESCRIPTION_OF_FARMER_OAK_AN_INCIDENT_WHEN_`

まで入力した後に，次文字として「`F`」を入力する時の様子を示している(注2)．

2.3 ハフマン符号（Huff）

文字の出現確率がわかっていれば，ハフマン符号が平均的に最短（平均操作回数が最小）である [5], [6]．ここでは，言語モデルとハフマン符号の組み合わせとする．言語モデルが多少異なることを除けば，Roark の方法 [11] と同様である．

ハフマン符号は符号語の並びは一般にアルファベット順ではない．このため，文字を符号語順に並べると図 2(c) のように文字の並びがばらばらになり，文字をアルファベット順に並べると図 2(d) のように選択候補文字が分散してしまう．いずれも見やすい表示ではない．

2.4 最適アルファベット順符号（GW）

最適アルファベット順符号 [12], [13] を用いると，前節に記したハフマン符号の表示に関する問題を解消できる（図 2(e)）．ただし，言語モデルと組み合わせた場合は，1 文字入力する毎に符号は変化するので，分割位置は毎回変わることになる．

最適アルファベット順符号の平均符号語長はハフマン符号に比べ多少増加（劣化）する．これは次のように解釈できる．文字盤の左端または右端にある文字であれば，確率が高い場合に 1 操作で選択できるように符号を作ることができる．しかしそれ以外の文字は，いくら確率が高い場合であっても，選択に少なくとも 2 操作必要となるのである．

2.5 例外を許す最適アルファベット順符号（GW+）

最適アルファベット順符号の平均符号語長がハフマン符号に対して増加するのを抑制する方法として，以下の改善策を提案する．

最も確率の高い 1 文字について，アルファベット順の制約をはずして 1 操作で選択できるようにし，それ以外の文字についてはその 1 文字を除いたアルファベットに対して最適アルファベット順符号を用いることとする．ただし，この方法を採用するかどうかは操作回数の期待値を計算して判断する．詳細な手順は以下のとおりである．

（1） 次文字の生起確率に基づき **GW** の符号を作り，次文字に関する操作回数の期待値を次式で求める．ただし，l_a は **GW** による文字 a の操作回数である．

$$\mathrm{E}[L]^{\mathrm{GW}} = \sum_{a \in A} P(a|s) l_a \qquad (1)$$

（2） 次文字の出現確率が最大となる文字 a_0 の符

(注2)：このテキストは次章の評価に用いた book27 の冒頭部分である．

図 3　評価用プログラムの表示画面

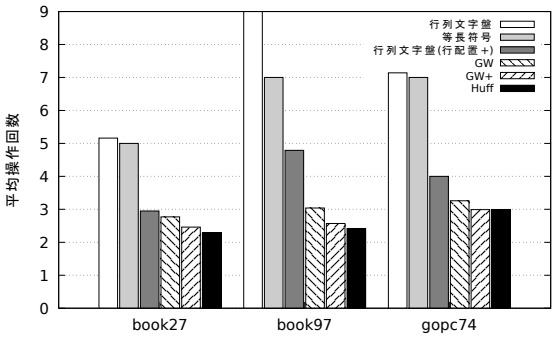


図 4　平均操作回数の比較

表 1　平均操作回数の比較

text (size)	book27 (729437)	book97 (763498)	gopc74 (851693)
行列文字盤	5.16	11.20	7.14
等長符号	5	7	7
行列文字盤（行配置 +）[10]	2.95	4.79	4.00
GW	2.77	3.04	3.26
GW$^+$	2.46	2.57	2.99
Huff	2.28	2.42	2.98

号語を「0」とし，それ以外の文字の符号語は先頭ビット「1」の後に a_0 を除くアルファベットに対する **GW** の符号語を続け，このようにして作った符号を **GW** $'$ とする．この場合の次文字に関する操作回数の期待値を次式で求める．ただし，l'_a は A から a_0 を削除したアルファベット A' に対する **GW** での文字 a の操作回数である．

$$\mathrm{E}[L]^{\mathrm{GW}'} = P(a_0) + \Big(1 + \sum_{a \in A'} P(a) l'_a\Big) \qquad (2)$$

（3） (1) と (2) の期待値を比較し，$\mathrm{E}[L]^{\mathrm{GW}'} < \mathrm{E}[L]^{\mathrm{GW}}$ ならば **GW** $'$ を採用し，そうでなければ **GW** を採用する．

図 2(f) の例では，次文字の生起確率は「0」について $P(\mathtt{0}|\mathtt{HEN\_})$ が最大で，$\mathrm{E}[L]^{\mathrm{GW}'} < \mathrm{E}[L]^{\mathrm{GW}}$ であったため，「0」が 1 回の操作で選択可能となっている．しかし，ここでは「0」ではなく「F」を入力するので，図のように「10110」の操作となった．2 操作目からは **GW** となるため，文字盤の候補文字群は左右に分割表示されている．

2.6　評価用プログラム

HTML および JavaScript 言語で符号化入力法の評価用プログラムを作成した(注3)．最適アルファベット順符号は，Garsia-Wachs アルゴリズムを用い，文献 [14] の付録 B を参考にして作成した．

ブラウザの表示画面を図 3 に示す．これは，図 2(f) で「F」の符号語を「110」まで操作した時の様子である．画面下に PPM の中で用いる文脈木を表示することもでき，文字を入力する毎に木が更新される様子を見ることができる．

Chrome ブラウザを使用した場合は外部スイッチを接続して操作することもできる．その場合は左に「0」右に「1」のスイッチを置くと画面の表示と対応する．

3.　操作回数の評価と考察

それぞれの符号化入力法について，シミュレーションにより平均操作回数を計算し，評価を行った．文脈 s における次文字 a の出現確率 $P(a|s)$ の計算には PPM（$o = 4$, Method C）[9] を用いた．事前学習は行っていない．評価用テキストは脚注参照(注4)．

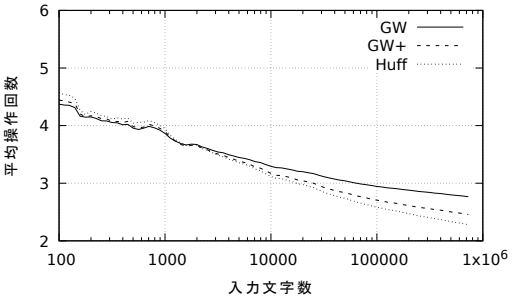
表 1 および図 4 は，book27, book97, gopc74 の 3 つのテキストについて，平均操作回数を示したものである．「行列文字盤（行配置 +）」は行列文字盤に予測文字を追加配置する方法 [10] で，参考のために記載した．

等長符号の平均操作回数が 5〜7，行列スキャン方式ではそれ以上であるのに対し，最適アルファベット順符号 **GW** の平均操作回数は 2.77〜3.26 で大幅に短縮された．例外を許す最適アルファベット順符号 **GW+** の平均操作回数は 2.46〜2.99 で，GW に対して更に 0.27〜0.47 の改善がみられた．この値はハフマン符号 **Huff** の平均操作回数 2.28〜2.98 に近い値（0.01〜0.18 の差）であった．

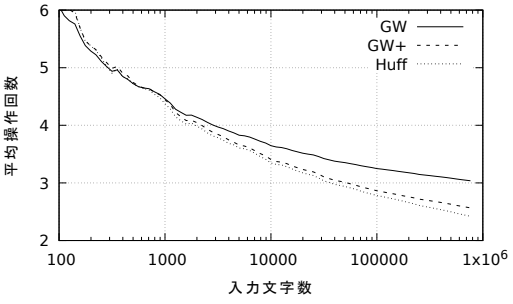
図 5 は入力文字数と平均操作回数の関係を 3 つの符号化法で比較したものである．どの評価用テキストについても 100000 文字を過ぎてもまだ収束しなかった．また，符号化入力法によ

(注3)：図 3 の評価用プログラムは以下のサイトに公開している．
http://siva.cc.hirosaki-u.ac.jp/usr/koyama/ppm/

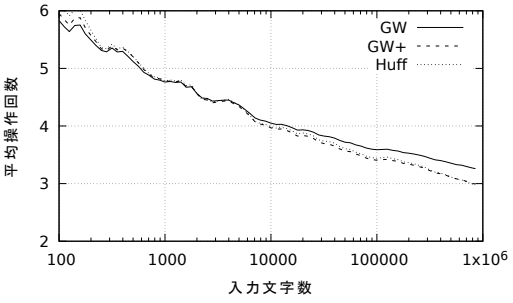
(注4)：book97 は Calgary Corpus [15] の book1（英語の小説）から制御コマンドを削除したものを用いた（$|A| = 97$）．book27 は book97 中の小文字をすべて大文字に変換し，A-Z 以外の文字を「_」に変換し，2 個連続する「_」を 1 個の「_」に置換したものである（$|A| = 27$）．gopc74 は原文（日本語の評論）をカナ文字列に直し，文節ごとにスペースを入れたものを用いた（$|A| = 74$）．これは，かな漢字変換のキーボード操作を模擬したものである．機械的に変換したものなので，一部不適切な変換結果も含まれているが，そのまま用いた．



(a)book27



(b)book97



(c)gopc74

図 5　入力文字数と平均操作回数

る性能の違いは 1000〜数 1000 文字入力した以降に徐々に鮮明になった．

Roark は，線形スキャン方式とハフマン符号化入力法を比較し，入力するテキストが言語モデルにマッチすると操作回数が少なくなり，方式による性能の違いは大きくなると述べている [11]．図 5 も，入力文字数が多くなると（よい言語モデルとなり），方式による差（最適アルファベット順符号とハフマン符号の差，あるいは最適アルファベット順符号と例外を許す最適アルファベット順符号の差）が大きくなるという同様の傾向を示している．

図 6 は，**GW+** で 1 操作文字を設けた場合に，実際にその文字が入力された採択率を示したもので，最終的には 60% 程度の高い割合でその文字が選ばれた．採択率は入力文字数とともに高くなり，これが図 5 でハフマン符号の性能に概ね追従していることに結びついていると考えられる．

4. おわりに

最適アルファベット順符号を用いた入力法および関連する入力法について，性能と文字盤表示の観点から検討した．

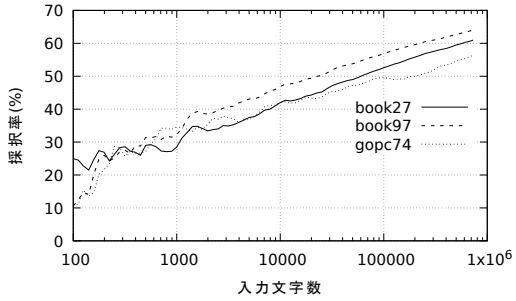


図 6　**GW+** における 1 文字目のヒット率

最適アルファベット順符号 **GW** は，文字盤はアルファベット順配列で 2 分割されるため見やすく，平均操作回数も行列スキャン方式や等長符号に対して大幅に改善された．更に操作回数の短縮を図りたい場合は，例外を許す最適アルファベット順符号 **GW+** を用いることにより概ねハフマン符号と同程度の性能が得られた．**GW+** は初回のみ文字盤の分割が左右の分割とならず変則的であるが，高率でその文字が採択されることを考えると，利用者にも受け入れられるのではないか考えられる．

データ圧縮や通信では圧縮率や計算量が大きな関心事となるが，本稿のような応用では「適切な制約下での最適符号」が重要な役割をはたす．最大符号語長制約のハフマン符号についても研究が進められている [6]．出現確率が非常に小さい文字であっても，操作回数が一定以上長くならないことは望ましいことであり，このような符号についても今後検討していきたい．

文　献

[1] P.A. Buhr, and R.C. Holte, “Some considerations in the design of communication aids for the severely physically disabled,” Medical & Biological Engineering & Computing, no.19, pp.725–733, 1981.

[2] R.I. Damper, “Text Composition by the Physically Disabled: A Rate Prediction Model for Scanning Input,” Applied Ergonomics, vol.15, no.4, pp.289–296, 1984.

[3] “Heraty Ladder,” http://takaki.la.coocan.jp/hearty/

[4] W.J. Crochetiere, G.S. Baletsa, and R.A. Foulds, “An Anticipatory Display for Communication by the Severely Disabled, Non-vocal person,” Society for Information Display DIgest, pp.150–151, 1977.

[5] D. Huffman, “ A method for the construction of minimum redundancy codes,” Proc. of the IRE, vol.40, No.9, pp.1098–1101, 1952.

[6] J. Abrahams （山本博資訳）, “ハフマン符号木に関連した話題,” 応用数理, vol.8, no.2, pp.4–20, 1998.

[7] S. Koyama, “An Adaptive Coding-Based Selection Scheme for a Communication Aid,” IEICE Trans. Fundamentals, vol.E73-A, no.11, pp.1542–1544, 1995.

[8] M. Baljko, and A. Tam, “Indirect Text Entry using One or Two Keys,” Proc. of the 8th ASSETS, pp.18–25, 2006.

[9] T.C. Bell, J.G. Cleary, and I.H. Witten, “Text compression,” Prentice-Hall, 1990.

[10] 佐々木大真, 小山智史, “行列スキャン入力方式に組み込む予測文字の配置の検討,” 信学技報, WIT99-25, pp.9-13,2000.

[11] B. Roark, R. Beckley, C. Gibbons, and M. Fried-Oken, “Huffman scanning: using language models within fixed-grid keyboard emulation,” Computer Speech and Language, no.27, pp.1212–1234, 2013.

[12] T.C. Hu, and A.C. Tucker, “Optimal computer search trees and variable-length alphabetical codes,” SIAM Journal on Applied Mathematics, 21, pp.514–532, 1971.
[13] A.M. Garsia, and M.L. Wachs, “A new algorithm for minimum cost binary trees, SIAM Journal on Computing,” vol.6, no.4, pp.622–642, 1977.
[14] J.C. Filliatre, “A functional implementation of the Garsia-Wachs Algorithm,” Proceedings of the 2008 ACM SIGPLAN, pp.91–96, 2008.
[15] “The Calgary Corpus,” http://corpus.canterbury.ac.nz/descriptions/#calgary