情 報 学 基 礎 57−17
自 然 言 語 処 理 136−17
（2000. 3. 22）

Unicodeを用いたN-gram索引の一実現方式とその評価

原田 昌紀 風間 一洋 佐藤 進也
NTT 未来ねっと研究所
東京都武蔵野市緑町 3-9-11

Unicodeベースの全文検索の実現方法について，索引づけ方式を中心に検討を行い，N-gramの長さを文字ブロックごとに可変とする索引づけ方式を提案する．提案した方式をWWWサーチエンジンに適用し，日本語テキストに適したパラメータを推定する．また，言語依存の処理を追加する方法について述べる．

An N-gram indexing method for Unicode based search engine

Masanori HARADA, Kazuhiro KAZAMA, and Shin-ya SATO
NTT Network Innovation Laboratories
3-9-11 Midori-cho, Musashino-shi, Tokyo

We investigate some indexing methods in order to implement an Unicode based full-text search engine and propose an N-gram indexing method that varies N-gram length per character block. With the proposed indexing method applied to our WWW search engine, we estimate parameters suitable for Japanese. We also describe some possible language dependent extensions.

1 はじめに

今日のインターネットの普及によるボーダーレス化に伴い，ソフトウェアの国際化が急速に進んでおり，それに応じて電子化された文書の形式も変化している．たとえば，HTML[1]やXML[2]などのインターネット関連の文書の規格では，さまざまな言語を考慮して，言語や文字符合化に関する情報が付加されている．これらの規格では，文字集合としてUnicode[3]を利用することが多いので，サーチエンジンにおいてもUnicodeをサポートすることが要求される．また，既存の文字集合はUnicodeに変換できるだけでなく，現在では文字の追加作業もUnicodeに対して行なわれているために，文書を一旦Unicode文字シーケンスに変換してから索引づけするのは妥当だと思われる．

そこで，本稿では，Unicode文字シーケンスを索引づけし，全文検索を実現する方法について検討する．

以下，第2節において，研究の背景と目的，

関連する研究を述べる．第3節では，可変長N-gramによるUnicode文字シーケンスの索引づけ方式を述べる．第4節では，WWWサーチエンジンに適用した結果を述べる．第5節では，提案する手法に言語に依存した拡張を追加する方法について議論する．最後に今後の展望を述べる．

2 研究の背景と目的

2.1 研究の目的

我々は，インターネット上に分散したサーバ上の情報を効率よく探索できるインフラストラクチャIngrid[4]の研究を進めてきたが，以下の問題点が存在した．

1. さまざまなプラットフォーム上で動作させるためにC言語で可搬性に注意して実装したが，それでも移植作業や動作確認作業が必要であった．
2. さまざまな言語で記述された情報を扱えるようにMule内部コードを採用したが[5]，HTMLやXMLなどのUnicodeをベースにした規格への対応が遅れた．
3. 日本語部分は日本語形態素解析を用いたが，頻繁に新語が誕生し，さまざまな表記で記述されるインターネット上の情報に対応できる辞書の保守が困難だった．
4. 他の言語に対応する際に，日本語形態素解析などの実装が難しい言語依存部を新たに用意する必要があった．

そこで，Ingridの後継である分散情報探索ツールキットPinot[6]では，分散情報探索アルゴリズムの改良作業と同時に，次の方針に従って全文検索モジュールJerkyを開発している．

1. プログラムを特定のプラットフォームに依存しないように設計し，Java言語で実装することで，移植作業を不要にする．
2. インターネット上の表記の揺れに比較的ロバストな検索手法を採用することで，保守作業を省力化する．
3. Unicodeを内部コードとして採用することで，HTMLやXMLとの親和性を高める．
4. 検索部分を特定の言語に依存しないように設計すると共に，言語ごとに異なる処理も追加できるように設計することで，検索効率向上のためのカスタマイズを可能にする．
5. 大量の文書情報に対応できるスケーラビリティを持たせる．

また，我々は最適な索引づけ方式は，検索対象となる文書の量，文書に用いられる言語，検索語における文字の生起頻度などによって異なると考え，パラメータの変更によって，さまざまな用途に対応可能な索引づけ方式を検討した．

2.2 関連研究

従来の全文検索の研究では，特定の言語をターゲットとした索引づけ方式が提案されるのが通例であった．特に中国語・日本語・韓国語・タイ語など，東アジア圏の言語では，一般にテキストが分かち書きされないため，さまざまな索引づけ方式が開発されている．

代表的な索引づけ方式としては，テキストを形態素に分割する方式，suffix arrayを作成する方式，N-gram単位に分割する方式がある．

テキストを形態素単位で索引づけする方式は，精度の高い検索を実現できることから，広く用いられている．しかし，言語ごとに形態素解析システムを用意しなくてはならない．

suffix array方式は，マルチバイト文字に対応可能であり，ストップワードを含む任意の文字列を高速に検索できるなどの利点を持つ[10]．しかし，サーチエンジンのように短い文字列での検索が大半を占める場合には，その利点が発揮されにくい．また，検索時にテキストにランダムアクセスする必要があるため，大規模な検索システムでは運用上のオーバーヘッドが大きくなる．

N-gram 方式では，テキストから一文字ずつずらした長さ N の文字列を切り出して，それらの生起位置情報を転置索引に記録し[1]，それらに対して隣接判定を行なうことで，辞書を必要としない検索を実現する．

近年は，N-gram の長さを可変とすることにより，検索速度を向上させる方法がいくつか提案されている [9][11]．また，N-gram 方式の変形として，文字の頻度情報を用いて，テキストを確率的に分割する方式や，辞書を用いた語単位の分割と N-gram 単位の分割を併用する方式も提案されている [12][13][14]，

N-gram 方式はすでに多くの言語でその有効性が認められており，英語などを対象とした単語単位の索引づけとも併用できるため，我々の方針と照らし合わせて，もっとも適当な索引づけ方式であると考えられる．そこで，本稿では N-gram 方式に基づいて，Unicode 文字シーケンスを索引づけする方法について検討する．

3 Unicode テキストの索引づけ

3.1 Unicode

Unicode は、Unicode Consortium が提唱している文字集合で、世界中から収集した多くの文字を 16bit 単位の最小構成要素である Unicode 文字で表現する規格である．Unicode は、ISO/IEC 10646 のベースとなっており，これらの規格の制定は，現在も協調して進められている．

2000 年 2 月現在の最新版は 3.0 であるが，本稿のシステムは Java2 が現在準拠している 2.1 を使用している．Unicode 2.1 は，次のような特徴を持つ．

1. 16bit 単位で意味が明確に決められている．ただし，必ずしも 1 文字が 16bit で表されるわけではなく，たとえば，サロゲートのように 2 つの Unicode 文字で 1 文字を表したり，1 文字を複数の Unicode 文字で合成することができる．

2. グリフではなく文字に対してコードポイントが割り当てられている．たとえば，中国，日本，韓国のグリフが比較的似ている漢字を統合し、同じコードポイントを与えている．これを Han Unification と呼び，これらの漢字集合を CJK Unified Ideographs と呼ぶ．

これらの特徴のために，Unicode 文字を操作する際には，次の点に注意しなければならない．

1. 同じ文字が異なる Unicode 文字シーケンスで表されることがある．文字列の同一性を判定するためには，正規化処理が必要である．

2. Unicode は，言語の区別をしないマルチスクリプト文字集合である．国際化や多言語処理を正しく行なうためには，扱う Unicode 文字シーケンスの言語情報を知る必要がある．

3.2 N-gram 方式の Unicode での実現

これまでに日本語テキストに対して，ひらがな，カタカナ，漢字といった字種の違いに応じて N-gram の長さを変化させる方式がいくつか提案されてきた [11][8][9]．

一方，Unicode に定義されている文字も，字種や関連した記号ごとにグループ化され，65 の文字ブロックに分類されている．そこで本稿では，既存の方式を拡張し，語単位で索引づけするか，どのような長さの N-gram 単位で索引づけするかといったパラメータの基準として，Unicode 文字ブロックを採用する．また，Unicode では文字に文字プロパティが付与されていることを利用し，記号等の文字カテゴリを判別して適切に処理する．

なお，ひらがな，カタカナ，漢字はそれぞれ，Hiragana，Katakana，CJK Unified Ideographs という文字ブロックに対応する．以下では，これらの文字ブロックに属する文字を「ひらがな」「カタカナ」「漢字」と表記する．

ただし，既存の方式では，次の 2 点について，その根拠が明確ではなかった．

[1]本稿では隣接する N 文字から構成される文字 N-gram のことを，単に N-gram と表記する．

1. N-gram の長さの決定方法

2. 複数の字種を含む N-gram の扱い

以下ではこれらについて考察を加え，Unicode へ適用する際の指針を述べる．

N-gram の長さの決定方法　N-gram による索引づけでは，索引ファイルを転置索引として構成するのが一般的である[2]．転置索引は，N-gram 自体を格納する語彙ファイルと，それらの生起位置情報を格納する参照ファイルに分けられる (図1)．

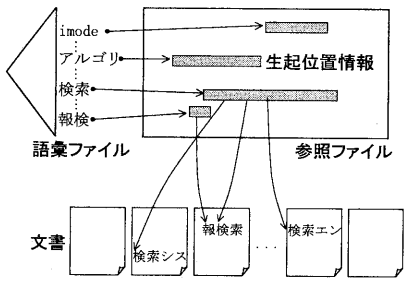

図 1: 転置索引の構成

生起位置について 1 つの N-gram を切り出す限り，生起位置情報の総数は，検索対象テキストのサイズのみで決定するので，参照ファイルのサイズは N によらず一定である[3]．

ただし，N を大きくすることで，N-gram 一つあたりの生起位置情報は減少する．また，少数の N-gram のみで検索文字列の位置が求められるため，ファイルから読み込む生起位置情報の量が減少し，隣接判定に要する時間が短縮される．

しかし，N-gram 数は指数関数的に増加するため，漢字のように文字数が大きい字種に対して，N を大きくすることは現実的でない．また，検索文字列の長さが N より短い場合には，その文字列を含むすべての N-gram の生起位置情報をマージする必要があり，検索速度が低下してしまう．

すなわち，文字ブロックごとの N の値は，その文字ブロックに含まれる Unicode 文字数による制限の範囲内で，語彙ファイルのサイズと検索速度のトレードオフを考慮して決めることになる．

複数の字種を含む N-gram の扱い　既存の方式の多くでは，字種が変化する部分では，境界をまたがる N-gram は許されておらず，単独の字種からなる長さ N 未満の N-gram が切り出される．このような方式を用いると，特に unigram が切り出される場合には，それを含む文字列の検索に非常に大きなコストがかかる．たとえば，「舞の海」や「最小 2 乗法」の検索が難しくなる．

この問題を避けるため，異なる文字ブロックに属する二つの文字が隣接している部分では，それらを bigram として切り出す．unigram として切り出す方式と比べると，語彙ファイルのサイズは若干大きくなるが，転置ファイルのサイズは変化しない．

3.3　索引づけの手順

N-gram 方式による索引づけは，以下の手順で行なう．

1. 文字シーケンスを正規化する．
2. 一部の文字群を空白に置換する．
3. 文字シーケンスを空白で分割し，得られた文字列をさらに N-gram に分割する．
4. N-gram の位置情報を転置索引に格納する．

正規化　Unicode 文字シーケンスの比較や索引づけには，正規化が必要となる．

表1に UTR#15[17] が規定している 4 種類の正規化形式 D，C，KD，KC を示す．ここで，Canonical Decomposition は，合成済文字 (Precomposed character) をそれと等価な文字シー

[2] シグネチャファイルを用いる方式では，正確な文字列検索ができないため，今日ではあまり用いられない．性能面でもメリットが小さいとの指摘がある [15]．

[3] 生起位置情報を符号化して格納する場合は，圧縮率の変化による差が生じる．

ケンスに可能な限り分解する操作であり，Canonical Compositionは逆に文字シーケンスから合成済文字を合成する操作である．たとえば，ö(U+00F6)は，o(U+006F)と ¨ (U+0308, COMBINING DIAERESIS)に等価である．

Compatibility Decompositionは，Canonical Decompositionに加えて，いわゆる半角カタカナなど，互換性維持のために重複して定義されている文字を，標準的な文字シーケンスに変換する．

表 1: UTR#15が定義する正規化形式

名称	概要
形式D	Canonical Decomposition
形式C	Canonical Decomposition + Canonical Composition
形式KD	Compatibility Decomposition
形式KC	Compatibility Decomposition + Canonical Composition

文字シーケンスの同一性判定は形式Cや形式Dで実現される．しかし，情報検索の用途では，互換性維持のための文字を区別する必要はほとんどなく，検索もれを減らす意味でも，Compatibility Decompositionを用いるのがよいと思われる．

ただし，N-gram方式では，濁音などが文字として分解されると，「コート」が「コート゛」にマッチするといった問題がある．そこで，形式KCを採用し，一度Compatibility Decompositionによって分割した後で，Canonical Compositionによって合成する．

なお，HTMLテキストを索引づけする際には，まず文字実体参照を対応するUnicode文字に置換してから正規化を行う．たとえば，`&`は&(U+0026, AMPERSAND)に，`&#x00A3;`は£(U+00A3, POUND SIGN)に置換する[1].

記号類の除去　正規化を行った後に，記号など検索文字列として使用する意味に乏しい文字を除去することで，索引のサイズを抑え，処理を簡潔にすることができる．そこで，Unicode2.0の規格において，文字プロパティがLetter あるいは Decimal digits となる文字のみを残し，それ以外の文字は空白に置き換える．

N-gram分割アルゴリズム　分割アルゴリズムは以下のようになる．図2に漢字はN=2，ひらがなはN=3，カタカナはN=4，Basic Latinは語単位に分割する例を示す．

手順1 入力文字列を異なる文字ブロックに属する文字が隣接する位置で分割し，N-gram候補とする．

手順2 手順1で得られたN-gram候補のリストのうち，分割可能であるN-gram候補を，1文字ずつずらしながら，文字ブロックごとに決められた長さのN-gramに分割し，元のN-gram候補と置き換える．N-gram候補の末尾は，N未満の長さで重複して切り出し，unigramになるまで分割する．

手順3 手順2で得られたN-gram候補のリストにおいて，長さ1のものがあれば，その一つ前のN-gram候補の末尾の文字と組み合わせたbigramを加える．また，その次のN-gram候補の先頭文字と組み合わせたbigramの二つを候補に加える．

検索文字列のN-gram分割　検索時には，まず索引づけ時と同じ手順によって，検索文字列をN-gramに分割した上で，以下の手順で処理を行う．

手順4 手順3までに求められたN-gram群から，検索文字列を被覆するN-gramの組み合わせのうち一つ選択する．一般的には，ディスクI/Oと隣接判定回数を最小にするため，生起位置数の和が最小になる組を求める[16].

手順5 生起位置情報の少ないN-gramから順に隣接判定を行ない，すべてのN-gramの判定を終えるか，検索結果が0件になるまで続ける．

ただし，最後の(もっとも右の)N-gramの長さが1であるときは，その文字を先頭

に持つすべてのN-gramの生起位置情報をマージして用いる．

入力 "iモード端末D502iを買いました"

手順1 ("i",0) ("モード",1) ("端末",4) ("D502i",6) ("を",11) ("買",12) ("いました",13)

手順2 ("i",0) ("モード",1) ("ード",2) ("ド",3) ("端末",4) ("末",5) ("D502i",6) ("を",11) ("買",12) ("いまし",13) ("ました",14) ("した",15) ("た",16)

手順3 ("iモ",0) ("モード",1) ("ード",2) ("ド端",3) ("端末",4) ("末D",5) ("D502i",6) ("iを",10) ("を買",11) ("買い",12) ("いまし",13) ("ました",14) ("した",15) ("た",16)

図 2: 索引づけ時のN-gram分割

4 WWWサーチエンジンへの適用

4.1 日本語に適したN-gram長の推定

現在，Jerkyを使用したWWWサーチエンジンODIN[18]を公開している．ODINは2000年2月の時点でJPドメインに存在する約597万URLのWebページ(HTMLファイル)を収集し，検索対象としている．

以下では3.2節で述べたトレードオフを考慮し，サーチエンジンで使用された検索語から，大量の日本語テキストを対象とする際に適したN-gramの長さを推定する．

1999年10月1日0時0分から，1999年10月31日23時59分までに使用された検索語85,697語に対して，3.3節で述べた正規化を行ない，得られた文字列において同じ文字ブロックに属する文字が連続する数を求めた．カタカナ，漢字，ひらがなの連続数を，それぞれ図5，図3，および図4に示す．点線は異なる文字ブロックに属する文字を含む検索文字列に限定して，連続数をカウントした場合である．

図3と図4から，カタカナ4文字を含む検索語，漢字2文字を含む検索語が多いことがわかる．したがって，カタカナは4-gram，漢字はbigram単位で分割すればよいと思われる．

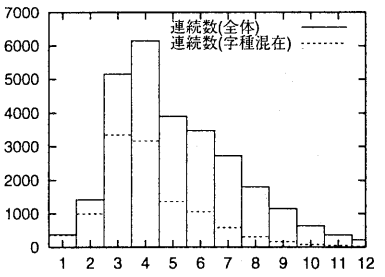

図 3: 検索語におけるカタカナの長さ

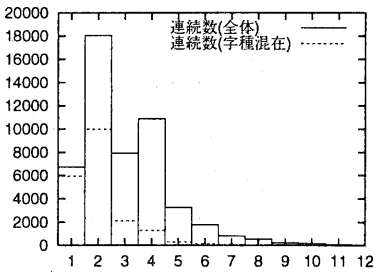

図 4: 検索語における漢字の長さ

また，図5のグラフによれば，ひらがなの連続数は1となることが多いが，ひらがなが1文字だけで用いられることはほぼ皆無であるため，本稿で提案する方式では他の字種と隣接したbigramとして検索される．そこで，ひらがなはtrigram単位の分割を基本とすればよいと推測される．

4.2 N-gramの頻度と索引サイズ

前節で述べたパラメータを用いて作成された転置索引において，語彙ファイル中のN-gram数と，参照ファイル中で生起位置情報が占める割合(占有率)を表2に示す．実際の索引では，Basic Latinのみからなる語や，少数のCJK Symbols and Punctuation, Greek, Cyrillic, Hangul Syllablesなどからなる N-gramが索引づけされているが，簡単のため省略する．この表から，異なる文字ブロックに属する二文字からなるbigramによる語彙ファイルサイズの増加は，[漢2]に比べれば十分に少ないことがわ

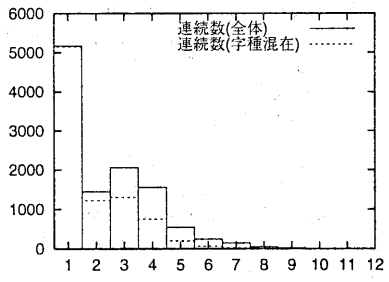

図 5: 検索語におけるひらがなの長さ

表 2: 転置索引の内訳 (抜粋)

種別	N-gram 数	平均生起位置数	占有率
漢 1	6756	32273	5.0 %
漢 2	2712107	347	21.5 %
漢 - 平	172231	3078	12.1 %
漢 - 片	168297	174	0.7 %
漢 - L	95320	334	0.002 %
平 1	86	2771285	5.4 %
平 2	6913	55051	8.7 %
平 3	308269	2478	17.4 %
平 - 漢	214607	1959	9.6 %
平 - 片	6730	8290	1.3 %
平 - L	3110	6201	0.4 %
片 1	92	694949	1.5 %
片 2	7058	23677	3.8 %
片 3	173046	896	3.5 %
片 4	1739162	164	6.5 %
片 - 平	6415	11449	0.8 %
片 - 漢	165054	201	1.7 %
片 - L	3196	1420	0.1 %
計	5788449	3614226	100 %

平:Hiragana, 片:Katakana, 漢:CJK Unified Ideographs
L:Basic Latin, 1:unigram, 2:bigram, 3:trigram, 4:4-gram

かる.

一方, これらを仮に unigram で切り出していたとすると, たとえば [平 - 漢],[平 - 片],[平 - L] の生起位置はすべて unigram として索引づけされ, [平 1] の平均生起位置数は表中の値の約 3.1 倍に増加する. しかも, ひらがな 1 文字を含むすべての検索語で, この膨大な生起位置情報が必要となり, 検索速度は著しく低下してしまう.

5 言語に依存した処理の追加

検索対象テキストと検索語に用いる言語があらかじめ特定される場面では, 言語ごとに異なった索引づけを行なうことで, 精度 (precision) や再現率 (recall), 処理性能を向上できる場合がある. 例をいくつか挙げる:

- 英語におけるステミングのように, 索引作成時と検索時に語句の正規化を行なうことによって, 検索もれを少なくできる.
- 中国語, 日本語や韓国語でも, N-gram 方式に加えて, 形態素解析を併用した索引づけをすることで, 検索精度を重視した検索ができる.
- ドイツ語でしばしば用いられる長い複合語を, 分解して索引づけできる.
- 言語ごとにストップワードのリストを用意すれば, 索引サイズが減少する.

本稿でのべた可変長 N-gram 方式は, 言語を特定せずに Unicode 文字シーケンスを索引づけするが, こうした言語に依存した索引づけと共存できる.

すなわち, N-gram 分割アルゴリズムの手順 1 および手順 2 を, 言語依存の分割アルゴリズムに置き換え, 正規化された語あるいは N-gram の生起位置情報を転置索引に記録すれば, 基本的な検索アルゴリズムや索引ファイルのデータ構造を変更する必要はない.

ただし, 検索結果を特定の言語を限定する機能が必要な場合は, 索引ファイルの構成に若干の変更が必要になる. これには二通りの方法が考えられる.

1. 索引ファイルを言語ごとに分割する.
2. 語彙ファイルに格納する N-gram ごとに言語情報を付与する.

これらは機能的には大きな違いはないが, 処理性能の観点からは, 検索対象を一つの言語に限定した検索が多い場合には前者が適しており, そうでない場合には後者が適していると考えられる.

6 おわりに

本稿では, Unicode 文字シーケンスを可変長の N-gram で索引づけする方式を提案した.

今後は，提案する方式を韓国語や中国語などに適用し，その有効性の評価を行いたい．また，ODINでは5節で述べた方針にしたがって，辞書を用いた日本語テキストの分割処理を追加し，精度と検索性能の向上を図っていく予定である．

参考文献

[1] D. Raggett, et. al.(eds.): "HTML 4.01 Specification," W3C Recommendation, 1999.

[2] T. Bray, et. al.(eds.): "Extensible Markup Language(XML) 1.0" W3C Recommendation 10-February-1998, 1998.

[3] The Unicode Consortium: "The Unicode Standard, Version 2.0," Addison-Wesley, 1996.

[4] P. Francis, et. al.: "Ingrid: A Self-Configuring Information Navigation Infrastructure," the 4th International World Wide Web Conference, 1995.

[5] S. Shimizu, et. al.: "A Framework for Multilingual Searching and Meta-information Extraction," INET'97, 1997.

[6] 佐藤, 他: "分散探索アルゴリズムの実装と評価," WIT'99, 1999.

[7] 学術情報センター編: "全文検索 — 技術と応用," 丸善, 1999.

[8] 菊地: "日本語文書用高速全文検索の一手法," 電子情報通信学会論文誌 D-I, Vol.J75-D-I, No.9, pp.836–846, 1992.

[9] 赤峯, 他: "高速全文検索のためのフレキシブル文字列インバージョン法," Proc. of Advanced Database Symposium '96, pp.35–42, 1996.

[10] G. H. Gonnet, et. al.: "New Indices for Text: PAT Trees and PAT Arrays," in W. B. Frakes, et. al.(eds.), *Information Retrieval: Data Structures & Algorithms*, Prentice Hall, 1992.

[11] 松井, 他: "高速テキスト検索エンジン," 情報処理学会デジタルドキュメント研究会報告 97-DD-7, pp.15–21, 1997.

[12] Y. Ogawa, et. al.: "Overlapping statistical word indexing: A new indexing method for Japanese text," Proc. of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.226–234, 1997.

[13] J. H. Lee, et. al.: "Using *n*-Grams for Korean Text Retrieval," Proc. of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.216–224, 1996.

[14] J. Y. Nie, et. al.: "On Chinese Text Retrieval," Proc. of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.225–233, 1996.

[15] I. H. Witten, et. al.: "Managing Gigabytes," Van Hostrand Reinhold, 1994.

[16] Y. Ogawa, et. al.: "Optimizing query evaluation in n-gram indexing," Proc. of the 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, poster, pp.367–368, 1998.

[17] M. Davis, et. al.: "Unicode Normalization," Unicode Technical Report #15, Revision 18.0, 1999. `http://www.unicode.org/unicode/reports/tr15/`

[18] "ODIN" `http://odin.ingrid.org/`