ユニバーサルなテキスト入力システムをめざして

増井 俊之 †

† 慶應義塾大学 環境情報学部
〒 252-8520 神奈川県藤沢市遠藤 5322

masui@pitecan.com

あらまし パソコンや携帯電話などで様々な日本語入力システムが利用されているが、機器によって入力方式がばらばらであるうえに必要以上に複雑な機能を持つものが多い一方、異なるシステム間で辞書を共有するといった基本的な機能さえ実現されていない。ユビキタスコンピューティング時代には、あらゆる属性の人間がいつでもどこでもテキスト入力や検索を行なえるようになっているべきであるが、現状のシステムはその要求を満たしていない。

我々は、計算機を初めて操作するような初心者からキーボード熟達者まで、あらゆるユーザがあらゆる場所で効率良く利用することができる予測型日本語入力システム「Lexiera」を開発し利用している。本論文では将来の入力システムが備えるべき特徴について考察し、Lexierra における実現手法について解説する。

Toward Universal Text Input Systems

Toshiyuki Masui†

†Faculty of Environment and Information Studies
Keio University
5322 Endo, Fujisawa, Kanagawa 252-8520

masui@pitecan.com

Abstract Although many sophisticated Japanese text input systems are available these days on personal computers and mobile phones, most of them are developed independently and they do not share the interface and the data in common. For example, we cannot use a single conversion dictionary on different systems. To solve the problem, we are developing a simple and universal text input system called *Lexierra*, which can be used on wide range of systems like mobile phones, PDAs, and personal computers. Lexierra supports efficient text composition through a simple look-up algorithm which uses dictionaries and other resources available on the Internet. In this paper, we discuss the basic requirements of text input systems in the ubiquitous computing age, and show how they are supported in various Lexierra implementations.

1 はじめに

パソコンや携帯電話などで毎日大量の日本語テキストが入力されており、将来は台所や風呂や街角などあらゆる場所で日本語テキストが利用されるようになると予想されるが、現在の情報機器上の日本語入力手法は複雑なものが多く、万人のための情報化の大きな障害になっている。

現在の日本語入力システムには以下のような問題がある。

- **システムが複雑**

  多くの日本語入力システムにおいて、沢山の機能に関連するキーやボタンが用意されており、かえって利用のための障害となっている。現在広く使われている Microsoft Windows では、いわゆる「106 キーボード」及びその後継キーボード[1]が標準的に利用されており、「半角/全角」「変換」「無変換」「カタカナ/ひらがな」という 4 個のキーが日本語入力のために用意されているが、これらのキーの意味は明確ではなく、充分活用されているとはいえない。標準的な日本語入力システムは、日本語入力専用のキーに加え、平仮名や「半角カタカナ」に変換するためのファンクションキーなども利用するようになっているが、普通のユーザが日本語を入力するために多数のキーの意味を理解して使いこなすことは難しい。携帯電話や PDA、スマートフォンでも事情は同様で、入力のために多くのキーやボタンが用意されており、キーの利用方法やモード切換に悩むことが多い。

- **機器により入力手法が異なる**

  パソコンで普及している入力手法と携帯電話で普及している入力手法は大きく異なっており、辞書も変換手法も異なっている。たとえばパソコン上で固有名詞を登録しても、携帯電話や PDA でそれを利用することはできない。現在の状況でも、利用方法が異なるすべての機器の入力手法に習熟して辞書や操作方法をカスタマイズすることは不可能であるが、将来の街角に設置されるサイネージや全く新しい装置のことを考えると状況はさらに悪くなると考えられる。

[1] http://ja.wikipedia.org/wiki/キー配列

2 パソコンの日本語入力システムの問題点

現在のパソコンの日本語入力システムは、以下のような理由により複雑で使いにくくなっていると考えられる。

- **連文節変換**

  パソコンの日本語入力システムではいわゆる「連文節変換」機能が標準であると認識されているが、連文節変換システムには多くの問題点がある。連文節変換システムの問題点は次節で詳しく考察する。

- **インライン至上主義**

  日本語入力システムはアプリケーションプログラムと密に結合して動作するべきだと考えられている。

  アプリケーションプログラム上での日本語入力は「インライン」で行なわれるべきだと一般に信じられている。エディタや表計算ソフトに日本語の入力を行なうとき、画面上の日本語テキストを入力したい場所で文字入力を行ないたいというのは自然なことであるが、文字入力機能をアプリケーションと一体化するためには文字入力システムとアプリケーションの間で多くの情報を共有する必要があり、実装において面倒なことが生じる。

  たとえば「`masui`」というキー入力を「増井」という文字列に変換したいとき、変換が確定していない状況ではどこかに「`masui`」という文字列を表示しておく必要があるが、文字のサイズはアプリケーションの状態に以上するため、以下のような情報共有が行なわれる必要がある。

  1. アプリケーションが文字を表示する場合は未確定文字列を入力システムがアプリケーションに渡す必要がある
  2. 入力システムが未確定文字列を表示する場合はアプリケーションの文字

サイズを入力システムが知る必要がある

これらに限らず、アプリケーションプログラムと入力システムの間で様々な情報共有が必要になる。アプリケーションの文字表示サイズやフォントに関して入力システムが面倒を見る必要はないし、あらゆる言語の入力システムに対してその内部状態をアプリケーションが把握するのは大変であろう。アプリケーションと入力システムの間での情報共有は最低限にするべきであろう。

日本語入力システムとアプリケーションシステムが競合する可能性もある。たとえば、日本語入力システムが「A」というショートカットキーを利用するようになっており、アプリケーションも「A」というキーを別の用途に利用していた場合、これらの間で調停が必要になる。あるプラットフォームにおいてこれらの整合性をとることができたとしても、別のプラットフォームではその方法が全く通用しないかもしれない。

- **特殊機能の氾濫**

  多くの日本語入力システムにおいて、「平仮名確定」「半角カタカナ確定」のようなショートカットキーが定義されているが、文字種ごとにこのようなキーを用意するとすると「ギリシャ文字変換」「丸数字変換」など際限なくキーが必要になってしまう。日本語入力にしか利用できない機能を多くのキーに割り当てることは好ましくない。

3 連文節変換の問題点

パソコン上では連文節変換による日本語入力方式が主流になっているが、連文節変換方式は以下のように多くの問題点がある。

- **正確な入力が必要**

  現在よく使われているほとんどの日本語入力システムでは、入力文字が 1 文字でも間違っていれば変換することができない。「ふいんき」を「雰囲気」に変換するようなシステムは存在するが、これは「雰囲気」を「ふいんき」という読みで登録しているだけであり、「ふんにき」を「雰囲気」に変換することはできない。

  連文節変換を行なう場合は長い文字列の読みを全く誤りなく入力する必要がある。パソコンのキーボードを正確に高速に操作できるユーザにとってはこれは大きな問題にはならないが、キーボードが使えない状況や、キーボード操作がうまくできないユーザにとっては大きな負担になる。

- **完全な読みの入力が必要**

  大抵の連文節変換システムでは、単語の読みが長い場合でも読みを全て入力する必要がある。たとえばローマ字モードで「品川駅」を入力するためには「shinagawaeki」という入力しなければならないが、現在携帯電話などで広く使われている POBox[2] のような予測式入力システムを使う場合は「shinag」程度入力したところで候補中から「品川駅」を選択することが可能である。連文節変換システムで「品川駅に行きました」を入力する場合は「shinagawaekiniikimashita」のような長い入力を全く間違えずに入力する必要がある。

- **変換誤りの訂正が必須**

  ユーザが入力したいと思っている文字列はユーザの頭の中にしか存在せず、読みの入力のみから計算することはできない。例えば「きょうはいしゃにいった」は「今日は医者に行った」「今日歯医者に行った」のどちらに変換すべきなのか計算機が判断することは不可能である。このため、長い文字列を正確に入力した場合でもシステムの提示した変換結果を修正する機能がどうしても必要になり、ユーザは余分なキー操作を強いられることになる。

- **非力なマシンで使えない**

  正確な変換を行なうためには充分な計算パワーが必要であり、小型の計算機では充分な速度で正確な連文節変換を行なうことができない。

このように見てみると、連文節変換システムは根本的に問題があるシステムだということができる。高度な自然言語処理による変換は技術的には挑戦的で面白いかもしれないが、ユニバーサル/ユビキタス時代の日本語入力手法としては不適当であるといわざるをえない。

4 入力システムの要件

これからのユビキタス社会/ネット社会においては、パソコン環境で熟練者にとってのみ使える入力手法ではなく、どこでも誰でも簡単に使えるシンプルで柔軟で効率良い入力方式が必要だと考えられる。

- キーやボタンなどの数が少ない
- 操作の種類が少ない
- 入力に必要な操作量が少ない
- ユーザがカスタマイズ可能
- 様々な環境で同じ方式が使える

このような要件を満たすためには、具体的には以下のような方式を採用することがよいと考えられる。

- **予測機能を用いて操作数を減らす**

  「shinag」というキー入力から「品川駅」を予測するなど、少ない操作でも文字入力が可能になるようにする。携帯端末や携帯電話では予測入力は一般的になっているが、これらで得られた知見をパソコンなどでにも適用する。

  現在の予測システムでは入力履歴、定型文、頻度情報などが利用されているが、様々なコンテクストやネット上の資源を利用することによってさらに効果的な予測を行なうことも可能であろう。

- **曖昧解消システムを用いて操作ミスを許容する**

  曖昧な入力や誤入力を許容するアルゴリズムを採用することにより、入力ミスに対応したり簡易入力を許したりする。

- **短い文節の変換を基本とする**

  予測システムを活用して「品川駅に行きました」という入力を行なう場合、「shinag」からの予測により「品川駅」を入力したり「ikim」からの予測により「行きました」を入力したりすることは容易であるが、「shinag」と「ikim」から予測を行なおうとすると、「品川に行きました」「品川で生きました」「品川区に行きました」など候補数が多くなり、多くの候補の中から目的の文を選択しなければならないことになって手間がかかると考えられる。候補数の爆発を防ぐため、文節ごとに入力を確定していく方法が望ましい。

- **単純な操作のみを提供する**

  パソコンの入力システムでは「カタカナ変換」のような機能のためにキーを割り当てているものが多いが、このような方針で機能を増やしていくと「ギリシャ文字変換」「丸文字変換」など無限に機能が増えてしまうことになってしまうので、このような付属的な機能は提供しない。具体的には、読みの入力、候補の選択、入力確定、訂正だけ機能を用意する。

- **アプリケーションとの連携を最小限にする**

  きめ細かな「インライン入力」を提供するのではなく、アプリケーションと入力システムが疎結合するようにする。

- **ネット上の資源を活用する**

  パソコンでも携帯端末でも同様の入力手法を採用し、辞書をネット上で共有することにすれば、ネット上の辞書を共通に利用することができるため、マシン毎に単語登録をする必要がなくなる。

ネット上で様々な方法で辞書を共有したり、ネット上のホットな単語を入力に利用したり、ネット上に存在するテキストを入力に活用したり、ネット上の資源を最大限に利用する。

- **共通言語で開発**

  ブラウザを利用する場合、JavaScript で入力システムを実装すれば様々なプラットフォームで共通に利用することができる。

5 Lexierra: ユニバーサルな入力システム

前述のような方針にもとづいて作成された日本語入力システム「Lexierra」を紹介する。Lexierra は様々な環境で実装されており、我々は様々なシステム上で毎日利用している。

5.1 実例 1: JavaScript で実装したブラウザ用入力システム

ブラウザ上で動作する日本語入力システムの例を図 1 に示す。エスケープキーを入力すると下部に日本語入力枠が表示され、ローマ字からの予測にもとづいて日本語入力を行なうことができる。ここでは「ny」を入力した結果として「入力」「入出力」などの単語が候補として提示されている。

図 1: JavaScript によるブラウザ上の入力システム

続いて「den」と入力したときの様子を図 2 に示す。「電話」「電車」のような候補に加え、 や のような画像が候補として提示されている。Lexierra では文字と画像を区別せず扱うことができるため、図 3 のような「画像かなまじり文」を簡単に作成することができる。

図 2: 画像を含む候補表示

図 3: 画像を入力してサイズ変更

5.1.1 辞書の構造

辞書データはサーバから図 4 のような JSON 形式[2]でブラウザに送られる。

辞書エントリの第 1 要素はローマ字の読み、第 2 要素は変換される文字または画像を示している。画像の場合は Gyazo[3] の ID になっている。また第 4 要素は単語のカテゴリを示しており、第 3 要素は単語がどのカテゴリに接続可能かを

[2] http://www.json.org/
[3] http://Gyazo.com/

```
[[".", "。", 0, 1000],
 [",", "、", 0, 1000],
 [".", "。", 0, 1001],
 [",", "、", 0, 1001],
 ...
 ["denwa",'04451e2727b87f79262179b178d8ecf5',0,1111],
 ["densha",'fdaa971e4471b58cd156008691e707f8',0,1111],
 ["densha",'fdaa971e4471b58cd156008691e707f8',0,1111],
 ...
 ["toukyou", "東京", 0, 1002],
 ...
 ["tabe", "食べ", 1, 1492],
 ...
 ["ru", "る", 1001, 1],
 ["ru", "る", 1000, 1],
 ["na", "な", 16, 1],
 ["re", "れ", 17, 1],
 ["re", "れ", 18, 1],
 ...
 ["reba", "れば", 0, 17],
 ["ru", "る", 6946, 17],
 ["ru", "る", 1001, 17],
 ...
]
```

図 4: 辞書データ

示している。「食べ」は下一段活用動詞の語幹であり、その後には「ます」「ない」のような助動詞が続く。この接続情報を利用して再帰的に辞書検索を行なうことにより、「食べます」のような活用語を検索して予測候補とすることが可能になっている。

このような情報を含む辞書は図 5 のように、Wiki システム「Gyazz[4]」の上で編集して作成している。単語の読みとカテゴリ名、接続可能なカテゴリ名をテキストで定義している。Web 上で編集を行なうことができるので、新しい単語を追加したり重みを変更したりすることが容易にできる。この Wiki データをもとにして図 4 のような JSON 形式の辞書データを生成している。

本システムを利用すると、JavaScript が動いて日本語が表示できるブラウザさえあれば日本語入力を行なうことができるため、海外のマシンで日本語を入力しなければならない場合などで非常に便利である。

[4]http://Gyazz.com/

5.2 実例 2: iPhone 用入力システム

iPhone やペン端末では独立したキーボードを利用することができないので実例 1 のような入力システムとは相性が悪い。実例 1 のシステムと同じ辞書や変換アルゴリズムを使用し、入出力部分を iPhone 用に特化したシステムを図 6 に示す。

読みの指定は五十音を利用する。「あ」キーをタッチすると図 7 のように「あ」行の文字がメニューとして表示されるので、目的の文字まで指をスライドさせて離すことにより「う」や「ぇ」などの文字を入力することができる。濁音や半濁音も同様である。

図 5: Gyazz 上の辞書編集

図 6: iPhone 用 Lexierra

「あ」「き」まで入力した状態を図 8 に示す。候補表示領域に余裕がある場合は「Google Suggest」へのアクセスを行ない、その検索結果も表示するようにしているため、辞書に登録されていない「秋月電子」のような固有名詞も候補としてリストされている。

iPhone のタッチパネルは複数の指を認識することができるので、「あ」を押してメニューを表示した後で指をスライドさせるかわりに、「あ」を押したものとは別の指で「お」を選択することもできるようにしている。

図 7: 「あ」をタッチしてメニューを表示

図 8: 「あ」「き」を続けて入力した状態

「あ」「か」などのボタンをタッチしてすぐに指を離すと、「あ]「い」「う」のような読みを正確に指定することなく、T9[5]と同様に、「あ行の文字」「か行の文字」のような曖昧な文字指定を行なうことができる。例えば「た」「あ」「か」「や」「あ」というキーを素早く押すと、「東京」「提供」などの候補を得ることができる。T9 は曖昧な文字指定で高速入力できるという特長を持っているが、逆に正確に読みを指定するのには手間がかかるという問題がある。我々の手法では、時間をかけると正確な読みを指定でき、素早く操作すると曖昧な読みを指定するという使い分けを行なうことができる。「二十一世紀」のような単語の場合、「にじゅううういっせいき」のように濁音/撥音/拗音を正確に入力しなくても「な」「さ」「や」「あ」「あ」「た」「さ」「あ」「か」とキーを押すだけで入力できる。

図 9: T9 と同様の連続子音入力により「東京」を得る

iPhone 版の Lexierra では、濁点をもたない「な」「ま」などを押したとき図 10 のようにメニューを利用して英数字や空白文字などの記号を選択することができるので、日本語/英語のようなモード切り換えが必要ない。このため必要なキーの総数はわずか 12 個であり、小さな

[5]http://www.t9.com/

入力画面しか持たない機器でも利用できる。

図 10: メニューによる英字入力

5.3 実例 3: Mac 用入力システム

図 11 は、実例 1 と同じ辞書とアルゴリズムを用いた MacOS 用の Lexierra である。MacOS 標準の入力ライブラリ (IMKit) を利用して Objective-C で作成しているが、辞書及び変換アルゴリズムは JavaScript のものと同じである。

図 11: Mac のメーラで日本語入力システムを利用

Mac 用 Lexierra は計算パワーに余裕があるため、[1] と似たアルゴリズムを利用して曖昧検索を可能にしている。このため図 12 のように「shimahawa」のような入力からでも「品川」が候補としてリストされる。

図 12: 曖昧検索

パソコン用の日本語入力システムはインライン変換をサポートしているものが多いが、Lexierra はインラインの編集をサポートしておらず、キャレットの位置近くに表示された入力ウィンドウの中で候補選択処理を行なうようになっている。連文節変換において文節単位を変更したりする場合はインライン処理が有効であるが、Lexierra のように短い文節単位で入力を行なう場合はインライン編集は不要である。

6 議論

我々は 2 年以上にわたって携帯端末やパソコンの上で日常的に Lexierra を利用している。日々の利用において気付いた点などを以下に述べる。

- **一般的利用感**

  我々はブラウザや Mac の上で日常的に Lexierra を利用しており、それ以外の日本語入力システムは利用していない。パソコンのブラウザや iPhone で利用する限り変換速度が問題になることはなく、快適に利用している。

  実例 1, 実例 3 のシステムはパソコン上での利用を想定したものであり、実例 2 のシステムは iPhone のような携帯端末上での利用を考えたものである。文字入力の使い勝手は全く異なるが、辞書も変換アル

ゴリズムも共通のものが利用されているため、候補が表示されたり候補を選択したりする感覚は共通である。高速に文字入力を行なうことができるパソコン上でも、それが不可能な携帯端末上でも全く同じシステムが利用できることから、それ以外の環境においてもユニバーサルに利用できると考えられる。

- **連文節変換システムとの比較**

  現在はパソコン上では連文節変換にもとづいて日本語入力を行なっているユーザが多い。連文節変換には「入力した読みがそのまま漢字を含む文章になる」というわかりやすさがあるため初心者にとって受け入れやすいうえに、現在まで多くのユーザに利用されてきた実績があるためだと考えられる。一方、携帯電話のような携帯機器では現在予測入力が主流になっている。我々は 10 年以上連文節変換機能を利用しておらず、パソコン上で Lexierra を利用することにより連文節変換より高速に日本語を入力している。この状況を考えると、熟練ユーザはパソコンでも携帯電話でも Lexierra のようなシステムを利用する方がデータ共有の意味でも入力効率化の意味でも望ましいと考えられるが、そのような認識が広まるには時間がかかるかもしれない。

- **ネット資源の活用**

  Lexierra の辞書はすべて共通でネット上で編集可能である。また実例 1、実例 3 の場合変換プログラム自体もネット上のものを利用するため、ネット上の資源を大きく活用しているシステムだということができる。

  これに加え、Google Suggest を併用することにより、入力できない固有名詞はほとんど無いといってよい状態になっている。特に、人気のある店やアーティストの名前を入力できない可能性は非常に低い。入力に利用可能性なネット上の資源は今後ますます増えると考えられ、その活用は意義があるといえるだろう。

- **動的辞書の利用**

  実例 1 のシステムは JavaScript で実装されており、辞書データは JavaScript の単純な配列になっているため、利用中に動的に辞書を変更することが可能である。たとえば、地名を入力する領域では地名辞書を選択し、人名を入力する領域では人名辞書を選択する、という具合にきめ細かく入力システムを調整することも可能である。

- **サーバとの連携**

  前述のように、実例 1 や実例 2 の Lexierra は自分の状態に応じてサーバの辞書を切り換えることができるが、変換作業に関連した情報をサーバに伝えることも可能である。単語学習、単語登録だけにとどまらず、操作に関連した各種の情報をサーバに送ることにより、よりきめ細かく入力を支援することが可能になる。

- **文法知識の利用**

  Lexierra の辞書は接続情報のみを利用した単純なものであるため、日本語以外の言語でも問題なく利用できる。実際、実例 1 で示したように、Lexierra は絵文字の入力にも有用であるため、絵文字的な言語に対しても問題なく利用できる。

- **連文節変換システムとの併用**

  現在広く利用されている連文節変換の技術を Lexierra と統合して利用することも考えられる。Lexierra は短い文節をもとにした入力作業に最適化されているため、長い読みを入力した場合は変換に失敗することがある。Lexierra で候補がみつからなかった場合、現在利用されているような連文節変換システムを起動し、その結果を候補として利用することにすれば Lexierra の利点と連文節変換の利点を同時に利用することが可能になると考えられ

る。実際、予測変換機能を搭載した一部の携帯電話の中にはバックエンドに Mobile Wnn のような連文節変換システムを搭載しているものがあり、両者の出力をマージして候補として表示している。

7　結論

パソコンでも携帯端末でもその他のユビキタス環境でも利用可能なユニバーサルな日本語入力手法「Lexierra」の紹介を行なった。なるべく多くの情報をネット上で共有し、ネット上の資源を最大限に活用するという方針にもとづいた Lexierra は将来のユビキタス環境においてあらゆる人間が効率良く入力を行なうためのベースとなるシステムとして有望である。Windows や Android などに対しても Lexierra を実装し、あらゆる機器において利用できるユニバーサルなシステムとして拡張を行なっていく予定である。

参考文献

[1] R. A. Baeza-Yates and G. H. Gonnet. Fast string matching with mismatches. *Information and Computation*, 108(2):187–199, February 1994.

[2] 増井 俊之. ペンを用いた高速文章入力手法. In 田中 二郎, editor, **インタラクティブシステムとソフトウェア** *IV:* **日本ソフトウェア科学会** *WISS'96*, pages 51–60. 近代科学社, December 1996.