クロスワードパズル生成問題の新しい定式化

学生番号　2013160
氏名　佐藤潤一
提出年度　平成 28 年度

目次

1.はじめに **1**
2.制約最適化問題 **3**
3.定式化 **4**
3.1 定式化の方針 4
3.2 表現の定義 4
3.3 変数 4
3.3.1 主要変数 4
3.3.2 補助変数 5
3.4 目的関数 6
3.5 制約 7
4.パズル生成実験 **12**
4.1 実験概要 12
4.2 実験 I 13
4.3 実験 II 15
5.応用可能性 **16**
5.1 スケルトンパズル自動生成への応用 16
5.2 観光振興への応用 17
6.まとめ **18**

謝辞 **19**
参考文献 **19**

付録 **20**

1. はじめに

本研究では**クロスワードパズル**を取り上げる。クロスワードパズルは白と黒で彩色された盤面と、カギと呼ばれるヒント文から成る。1 つの行もしくは列において、連結した 2 つ以上の白マスの極大系列をスロットと呼ぶ。カギを手掛かりにして全てのスロットに適切な語を割り当てることができれば、パズルは完成である。

クロスワードパズルは単なる娯楽にとどまらず、様々な分野から注目されている[1]。例えば教育の分野では、初等教育や英語教育を目的としたクロスワードパズルが存在する[2,3]。また、観光の分野では、ご当地の単語を含んだクロスワードパズルが存在する[4]。紙とペンで遊ばれてきたクロスワードパズルだが、近年のスマートフォンの普及に伴い、今後も様々な分野で活用されるものと考えられる。

本研究ではクロスワードパズルを、コンピュータを用いて自動生成することを考える。クロスワードパズルを特定の目的のもとで利用する場合、継続的な利用が求められることがある。例えば教育を目的とする場合、初等教育の 1 単元での使用にとどまらず、複数単元での継続的な利用が求められることが考えられる。また、観光振興を目的とする場合は、トレンドの変化に合わせてその都度新しいパズルを制作する必要がある。パズルの性質上、パズルの継続的な利用を実現するためには、異なるパズルを大量に生産する必要がある。一般的にパズルを人力で制作するには手間がかかるため、パズルを簡単に制作する方法が求められる。例えばパズル中で使用したい語の集合を用意しておき、それをコンピュータに与えるだけでパズルが自動生成されるような仕組みがあれば、パズル制作の手間を大きく軽減することができると考えられる。

コンピュータを用いてクロスワードパズルを自動生成しようという試みは、従来から研究されている。従来手法では**探索問題**や**制約最適化問題**に基づいた定式化を行い、アルゴリズムを適用することでそれを解く。例えば[5,6]では、入力として辞書と呼ばれる語の集合と、白もしくは黒に彩色された盤面が与えられ、辞書に含まれる語のみを用いて全てのスロットを埋めることを問う探索問題として取り扱われている。従来手法で生成できるのは、語の割当が完了した状態の盤面であり、カギについては何らかの方法で用意しなければならない。従来手法の問題点として、入力される辞書のサイズが小さい場合は解が存在しない可能性がきわめて高いために生成することが困難であること、入力として彩色された盤面が必要であることなどが挙げられる。教育や観光振興など、特定の用途をもつパズルを生成しようと考えたときは、辞書に含むことができる単語の数が限られることがある。また、解が存在するような盤面を決めることは一般に困難であるため、従来手法は特定の用途をもつパズルの生成に適さない。彩色された盤面の入力を必要とせず、辞書と盤面の大きさのみを入力としてクロスワードパズルを生成するソフトウェア[7]も存在するが、パ

ズルを生成する際に黒マスは連続しないこと、全ての白マスは連続していることなどの日本語クロスワードパズルの慣例[8]や、周囲 3 マスが黒マスであるマスが存在しないという独自のルールに従っているため、こちらも辞書サイズが小さい場合には、そもそも解が存在せず、パズルを作ることができないということが頻繁である。

以上を踏まえて本研究では、辞書サイズが小さい場合であってもパズルの生成を可能とするような、クロスワードパズル生成問題の新しい定式化を提案する。具体的には、辞書と盤面の 1 辺の長さnを与えるだけで大きさ$n \times n$の盤面の彩色と語の割当を同時に行い、必ず何らかのパズルを出力するような制約最適化問題に定式化するのである。提案手法では日本語クロスワードパズルの慣例に厳密に従うことはせず、多少慣例に背いてでもパズルを出力することを優先する。また、従来手法と同様に、語の割当が完了した状態の盤面のみを生成する。提案手法では、彩色された盤面の入力が不要になるという点で、従来よりも簡単にクロスワードパズルの自動生成が可能になる。また、辞書サイズが小さい場合でも必ず何らかのパズルを生成するため、辞書サイズが限られる場合でもパズルの生成が可能である。提案手法を用いてパズル生成実験を行ったところ、従来手法では解が存在しないような入力を与えても何らかのパズルを生成した。また、サイズの小さい辞書を用いて 1 辺の長さnが小さいパズルを生成する場合は、短い計算時間で最良解の更新が止まることがわかった。

以下、2 章で制約最適化問題について説明し、3 章で定式化の提案を行う。続く 4 章で実験を行い、5 章で提案手法の応用を検討する。最後に 6 章でまとめを行う。

2. 制約最適化問題

本章では制約最適化問題について説明する。本論文ではクロスワードパズルの生成問題を制約最適化問題として定式化する。制約最適化問題の一般形を以下に示す。

maximize	$f(x)$	
subject to	$g_j(x) \in C_j$	$(j = 1, \ldots, m)$
	$x_i \in D_i$	$(i = 1, \ldots, n)$

この問題では目的関数$f(x)$を最大化するが、その制約として関数$g_j(x)$の値域は集合C_jであることと、変数x_iの定義域は集合D_iであることが課される。

制約最適化問題はオペレーションズ・リサーチや人工知能における重要な基本問題の 1 つであり、一般には ***NP*困難**と呼ばれる複雑な問題のクラスに属する。制約最適化問題を解くための様々な厳密／近似アルゴリズムが知られており、コンピュータで解を求めるためのソルバーと呼ばれるソフトウェアも数多く存在する。厳密解を求めるソルバーで解く場合は、全探索を行うと計算時間が膨大になることが予想されるため、計算時間に上限を定めて、時間内に得られた解のうち最良のもので評価するのが一般的である。

3. 定式化

クロスワードパズル生成問題を、制約最適化問題として定式化し、それを解くアルゴリズムを適用することによって、自動生成を実現する。この章では、はじめに定式化の方針を説明し、表現の定義、使用する変数、目的関数、制約の順に説明する。

3.1　定式化の方針

定式化の方針は、盤面の 1 辺の長さnと辞書の入力に対して、辞書に含まれる全ての単語の先頭と最後尾に黒マスを表す文字■を追加した上で、$(n+2) \times (n+2)$のサイズに拡張した盤面に語を配置するというものである。拡張した盤面から両端の行と列を除いた$n \times n$の中央部分が、最終的に生成されたパズルとなる。

3.2　表現の定義

以下で使用する表現を定義する。任意の自然数$p, p'(p < p')$について、$[p] = \{1, \dots, p\}$、$[p, p'] = \{p, \dots, p'\}$とする。任意の$i, j \in [n+2]$について、$i$行$j$列のマスを$(i, j)$とする。与えられた辞書$D$を$m$語の集合$D = \{w_1, \dots, w_m\}$とする。それぞれの語$w_k \in D$は$l(k)$個の文字の系列$w_k = (w_{k,1}, \dots, w_{k,l(k)})$である。文字の集合を$A = [q] \cup \{■\}$とすると、上記の方針より$w_{k,1} = w_{k,l(k)} = ■$、$w_{k,2}, \dots, w_{k,l(k-1)} \in [q]$であり、本来の語$w_k$は$(w_{k,2}, \dots, w_{k,l(k-1)})$である。

3.3　変数

提案する定式化では 10 種の変数を使用する。その内訳は、3 種の主要変数と 7 種の補助変数である。以下でそれぞれ説明する。

3.3.1　主要変数

定式化全体を通して使用する変数を主要変数として定義する。主要変数として、単語の割当を表す変数x, yと、文字の割当を表す変数zを導入する。

変数$\{x_{k,i,j}\}$は語の横向きの配置を表す 0-1 変数である。$w_k \in D$、$(i, j) \in [2, n+1] \times [n+3-l(k)]$について、次に示すような値を取る。

$$x_{k,i,j} = \begin{cases} 1 & (i,j)\text{に語}w_k\text{の先頭を割り当て、横向きに配置する,} \\ 0 & \text{上記以外.} \end{cases}$$

変数$\{y_{k,i,j}\}$は語の縦向きの配置を表す 0-1 変数である。$w_k \in D$、$(i, j) \in [n+3-l(k)] \times$

$[2, n+1]$ について、次に示すような値を取る。

$$y_{k,i,j} = \begin{cases} 1 & (i,j)\text{に語}w_k\text{の先頭を割り当て、縦向きに配置する,} \\ 0 & \text{上記以外.} \end{cases}$$

変数$\{z_{a,i,j}\}$は文字の割当を表す 0-1 変数である。$a \in A$、$(i,j) \in [n+2] \times [n+2]$について、次に示すような値を取る。

$$z_{a,i,j} = \begin{cases} 1 & (i,j)\text{に文字}a\text{を割り当てる,} \\ 0 & \text{上記以外.} \end{cases}$$

3.3.2 補助変数

一部の制約や目的関数を設定する際に使用する補助変数を導入する。

変数$\{X_{i,j}\}$は(i,j)で語の交差が発生しているか否かを表す 0-1 変数である。$(i,j) \in [2, n+1] \times [2, n+1]$について、次に示すような値を取る。

$$X_{i,j} = \begin{cases} 1 & (i,j)\text{で語の交差が発生している,} \\ 0 & \text{上記以外.} \end{cases}$$

変数$\{u_{i,j}\}$は(i,j)が白マスか否かを表す 0-1 変数である。$(i,j) \in [n+2] \times [n+2]$について、次に示すような値を取る。

$$u_{i,j} = \begin{cases} 1 & (i,j)\text{が白マスである,} \\ 0 & \text{上記以外.} \end{cases}$$

変数$\{e^{\rightarrow}_{i,j}\}$は$(i,j)$とその右側に隣接する$(i,j+1)$の 2 マスについて、2 マス中の白マスの個数を示す変数である。$(i,j) \in [2, n+1] \times [2, n]$について、次に示すような値を取る。

$$e^{\rightarrow}_{i,j} = \begin{cases} 2 & (i,j)\text{と}(i,j+1)\text{がともに白マスである,} \\ 1 & (i,j)\text{と}(i,j+1)\text{のうちどちらか一方のみが白マスである,} \\ 0 & \text{上記以外 .} \end{cases}$$

変数$\{e^{\downarrow}_{i,j}\}$は(i,j)とその下側に隣接する$(i+1,j)$の 2 マスについて、2 マス中の白マスの個数を示す変数である。$(i,j) \in [2, n] \times [2, n+1]$について、次に示すような値を取る。

$$e_{i,j}^{\downarrow} = \begin{cases} 2 & (i,j)と(i+1,j)がともに白マスである, \\ 1 & (i,j)と(i+1,j)のうちどちらか一方のみが白マスである, \\ 0 & 上記以外. \end{cases}$$

変数$\{b_{i,j}^{\rightarrow}\}$は$(i,j)$とその右側に隣接する$(i,j+1)$の 2 マスについて、2 マスともに黒マスか否かを示す 0-1 変数である。$(i,j) \in [2,n+1] \times [2,n]$について、次に示すような値を取る。

$$b_{i,j}^{\rightarrow} = \begin{cases} 1 & (i,j)と(i,j+1)がともに黒マスである, \\ 0 & 上記以外. \end{cases}$$

変数$\{b_{i,j}^{\downarrow}\}$は(i,j)とその下側に隣接する$(i+1,j)$の 2 マスについて、2 マスともに黒マスか否かを示す 0-1 変数である。$(i,j) \in [2,n] \times [2,n+1]$について、次に示すような値を取る。

$$b_{i,j}^{\downarrow} = \begin{cases} 1 & (i,j)と(i+1,j)がともに黒マスである, \\ 0 & 上記以外. \end{cases}$$

変数$\{c_{v,v'}\}$はマスvとマスv'について、vとv'が隣接していてかつ 2 マスともに白マスのときはn^2、1 マスだけが白マスもしくは 2 マスともに黒マスのときは 0。マスvとマスv'が隣接していないときは 1 となる変数である。

$$c_{v,v'} = \begin{cases} n^2\left(max\left\{e_{i,j}^{\rightarrow}, 1\right\} - 1\right) & (v,v') \in \left\{\left((i,j),(i,j+1)\right),\left((i,j+1),(i,j)\right)\right\}のとき, \\ n^2\left(max\left\{e_{i,j}^{\downarrow}, 1\right\} - 1\right) & (v,v') \in \left\{\left((i,j),(i+1,j)\right),\left((i+1,j),(i,j)\right)\right\}のとき, \\ 1 & 上記以外. \end{cases}$$

3.4　目的関数

目的関数として以下の 3 つを考える。これらのうち任意のものを選んで使用することが可能である。

・目的関数 1：白マス数の最大化

白マス数を最大化する。以下に式を示す。

$$\text{maximize} \sum_{(i,j) \in [2,n+1] \times [2,n+1]} u_{i,j} .$$

・目的関数 2：交差数の最大化

語の交差が起きているマスの数を最大化する。次頁に式を示す。

$$\text{maximize} \sum_{(i,j) \in [2,n+1] \times [2,n+1]} X_{i,j} .$$

・目的関数 3：黒マスの連続にペナルティを課した上で白マス数を最大化

白マス数を最大化するが、黒マスの連続にはペナルティを課す。ペナルティには重みwを設定することができ、wの値によってペナルティの影響を調整することができる。以下に式を示す。

$$\text{maximize} \sum_{(i,j) \in [2,n+1] \times [2,n+1]} u_{i,j} - w \times \left(\sum_{(i,j) \in [2,n+1] \times [2,n]} b^{\rightarrow}_{i,j} + \sum_{(i,j) \in [2,n] \times [2,n+1]} b^{\downarrow}_{i,j} \right) .$$

3.5　制約

提案する定式化では変数間制約に加えて 7 種の制約を設定する。制約はそれぞれ、回数制約、外周制約、隣接制約、割当制約、語交差制約、文字一致制約、連結制約である。

・変数間制約

変数の間に成り立たなければならない関係を制約として設定する。8 種の変数間制約を設定する。$(i,j) \in [2,n+1] \times [2,n+1]$について次の式が成立しなければならない。

$$u_{i,j} + z_{\blacksquare,i,j} = 1 ,$$
$$X_{i,j} \leq u_{i,j} ,$$
$$X_{i,j} \leq u_{i,j-1} + u_{i,j+1} ,$$
$$X_{i,j} \leq u_{i-1,j} + u_{i+1,j} .$$

$\{e^{\rightarrow}_{i,j}\}$と$\{b^{\rightarrow}_{i,j}\}$については、$(i,j) \in [2,n+1] \times [2,n]$について次の式が成立しなければならない。

$$e^{\rightarrow}_{i,j} = u_{i,j} + u_{i,j+1} ,$$
$$b^{\rightarrow}_{i,j} = max\left\{ z_{\blacksquare,i,j} + z_{\blacksquare,i,j+1} - 1, 0 \right\} .$$

$\{e^{\downarrow}_{i,j}\}$と$\{b^{\downarrow}_{i,j}\}$については、$(i,j) \in [2,n] \times [2,n+1]$について次の式が成立しなければならない。

$$e^{\downarrow}_{i,j} = u_{i,j} + u_{i+1,j} ,$$

$$b_{i,j}^{\downarrow} = max\left\{z_{■,i,j} + z_{■,i+1,j} - 1, 0\right\}.$$

・回数制約

1 つの語は、多くとも 1 つのスロットにしか配置できないという制約である。同じ単語が同一パズル内に複数存在することを禁止する。$w_k \in D$について次の式が成立しなければならない。

$$\sum_{(i,j)\in[2,n+1]\times[n+3-l(k)]} x_{k,i,j} + \sum_{(i,j)\in[n+3-l(k)]\times[2,n+1]} y_{k,i,j} \leq 1\,.$$

・外周制約

サイズ拡張後の盤面において、端に存在する全てのマスには黒マスを表す文字■が割り当てられるという制約である。最終的にパズルとして取り出される部分は$(i,j) \in [2,n+1] \times [2,n+1]$であるため、端のマスに文字が入ることによる単語の分断を避ける。$i,j \in \{1,n+2\}$について次の式が成立しなければならない。

$$z_{■,i,1} = \cdots = z_{■,i,n+2} = 1\,,$$
$$z_{■,2,j} = \cdots = z_{■,n+1,j} = 1\,.$$

・隣接制約

$(i,j) \in [2,n+1] \times [2,n+1]$において$(i,j)$に隣接する全てのマスが黒マスの場合は$(i,j)$も黒マスにならなければならない。黒マスに囲まれたマスに不適切な文字が入ることを禁止する。$(i,j) \in [2,n+1] \times [2,n+1]$について次の式が成立しなければならない。

$$z_{■,i-1,j} + z_{■,i+1,j} + z_{■,i,j-1} + z_{■,i,j+1} - 3 \leq z_{■,i,j}\,.$$

・割当制約

全てのマスに、■を含む何らかの文字を割り当てなければならない。$(i,j) \in [n+2] \times [n+2]$について次の式が成立しなければならない。

$$\sum_{a\in A} z_{a,i,j} = 1\,.$$

・語交差制約

(i,j)に■が割り当てられていて、$(i,j+1)$, $(i,j+2)$に■以外の文字が割り当てられてい

る場合、(i,j)には横方向に語の先頭が割り当てられなければならない。図 1 の左側の状態では(4,3)から横方向に該当箇所が存在する。この場合、(4,6)に割り当てられる文字によっては(4,3)から横方向に連続する文字列が辞書に存在しないものになり得る。そこで、(4,3)から横方向に、辞書に存在する語を割り当てることで、辞書と矛盾しないようにする。縦方向についても同様である。横方向について考えるとき、$(i,j) \in [2,n+1] \times [n-1]$について、次の式が成立しなければならない。

$$-z_{\blacksquare,i,j} + z_{\blacksquare,i,j+1} + z_{\blacksquare,i,j+2} + \sum_{w_k \in D: j \leq n+3-l(k)} x_{k,i,j} \geq 0 .$$

また、縦方向について考えるとき、$(i,j) \in [n-1] \times [2,n+1]$について、次の式が成立しなければならない。

$$-z_{\blacksquare,i,j} + z_{\blacksquare,i+1,j} + z_{\blacksquare,i+2,j} + \sum_{w_k \in D: i \leq n+3-l(k)} y_{k,i,j} \geq 0 .$$

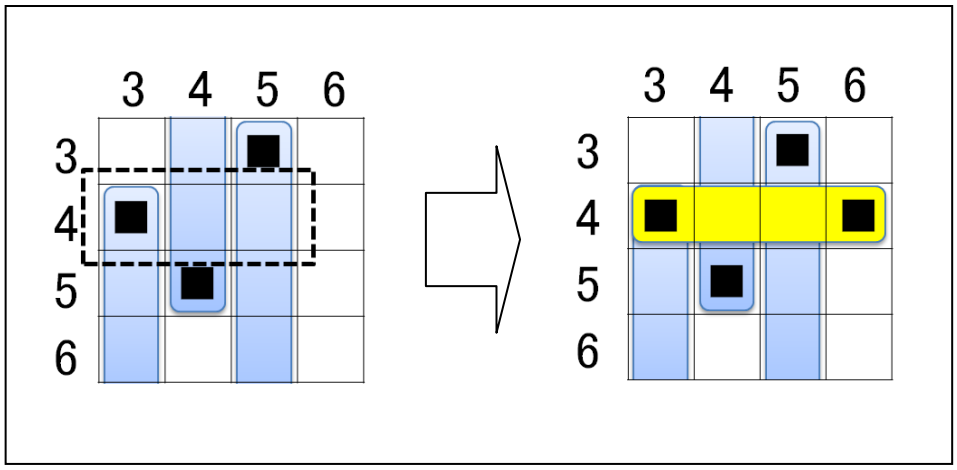


図 1 語交差制約

・文字一致制約

(i,j)に対して、単語w_kの先頭が横向きに割り当てられている場合、$(i,j),(i,j+1),\dots,(i,j+l(k)-1)$には、$w_{k,1},w_{k,2},\dots,w_{k,l(k)}$が割り当てられなければならない。提案する定式化では語の割当と文字の割当を別々の変数で行っているため、制約による対応付けが必要である。語の横向きの割当と文字の関係を考えるとき、$(i,j) \in [2,n+1] \times [n+3-l(k)]$、$w_k \in D$、$d \in \{0,\dots,l(k)-1\}$について次の式が成立しなければならない。

$$-x_{k,i,j} + z_{w_{k,d+1},i,j+d} \geq 0 .$$

また、語の縦向きの割当と文字の関係を考えるとき、$(i,j) \in [n+3-l(k)] \times [2,n+1]$、$w_k \in D$、$d \in \{0,\dots,l(k)-1\}$について次の式が成立しなければならない。

$$-y_{k,i,j} + z_{w_{k,d+1},i+d,j} \geq 0 .$$

・連結制約

すべての白マスは連結していなければならないという制約である。

パズルをグラフ$N = (V,E)$とみなし、**最大フロー問題**を応用することで定式化を行う。Vは節点の集合であり、パズルの盤面中に存在する$n \times n$個のマスと盤面の外に存在するマスtを合わせたn^2+1個のマスを節点として含む。また、Eは枝の集合であり、$(2,2)$から隣接する$(2,3), (3,2)$に出る辺 2 本と、$(2,2)$とtを除いた全てのマスvについて、vから隣接するマスv'に出る辺(v,v')とv'からvに入る辺(v',v)の総数$4n(n-1)-4$本、$(2,2)$とtを除いた全てのマスvからマスtに出る辺(v,t)の総数n^2-1本を枝として含む。以降、$(2,2)$をsとする。

sからフローを流し、Eに含まれる全ての枝をフローが通る。ただし、(v,v')を通ることができるフローの数には容量$c_{v,v'}$が設定されており、それを超えてはならない。sから出るフローの量とtに入るフローの量が等しいとき、パズル中の全ての白マスは連結しているという考えの元、定式化を行う。以下で定式化を示す。

初めにsを白マスに固定する。

$$u_{2,2} = 1 .$$

$v \in V \setminus \{s,t\}$について$v$と、$v$に隣接する節点$v'$との間に存在する枝$(v,v')$に流れるフローの量を$f_{v,v'}$とする。$f_{v,v'}$は$(v,v')$の容量$c_{v,v'}$を超えてはならない。$v \in V \setminus \{s,t\}$について次の式が成立しなければならない。

$$f_{v,v'} \leq c_{v,v'} .$$

また、$v \in V \setminus \{s,t\}$について、$v$に隣接する全ての節点から$v$に入るフローの量と、$v$から$v$に隣接する全ての節点に出るフローの量は等しくならなければならない。$v \in V \setminus \{s,t\}$について次の式が成立しなければならない。

$$\sum_{v' \in In(v)} f_{v',v} = \sum_{v' \in Out(v)} f_{v,v'} .$$

sとtについて、sから出るフローの量とtに入るフローの量が等しくならなければならない。

$$\sum_{v \in Out(s)} f_{s,v} = \sum_{v \in In(t)} f_{v,t} .$$

以上の定式化のもとで実行可能解が求められた場合、その解から生成されるクロスワードパズルの盤面上に存在する全ての白マスは連結している。しかしながら、この制約は非常に強い制約であり、辞書によっては生成されるクロスワードパズルの質が悪いものになり得るため、プログラムのパラメータにより採用するか否かを選択できるような仕様にしている。

4. パズル生成実験

2 章で定式化した制約最適化問題を IBM ILOG CPLEX CP Optimizer(ver. 12.6.2、以下 CPLEX)[9] を用いて解く。CPLEX は Gurobi Optimizer と並ぶ世界最高峰の数理最適化ソルバーの一つである。C++を用いて CPLEX のライブラリを呼び出すプログラムを作成した。プログラムの総行数は約 800 行である。実験に使用したデスクトップ PC の CPU は Intel® Core™2 Duo CPU E7600(クロック周波数は 3.06GHz)であり、メインメモリの容量は 4GB である。

本実験は実験 I と実験 II から構成される。実験 I では、提案手法とクロスワードギバー[7]に対して辞書と盤面の大きさを入力し、何らかのパズルが出力されるかを見る。その結果、実験 I で与えた全ての入力に対して、提案手法では何らかのパズルを出力したが、クロスワードギバーではいかなるパズルも出力しなかった。実験 II では、提案手法について、1 辺の大きさnに関わらず一律 3,600 秒の計算時間を設定した上で、最良解が求められる時間を見る。その結果、サイズが小さい辞書を用いて 1 辺の長さnが小さいパズルを生成する場合は、短い計算時間で最良解の更新が止まることがわかった。

4.1 実験概要

実験では、提案手法とクロスワードギバーを用いる。クロスワードギバーはシェアウェアで、辞書と盤面の大きさ$n \times m$を与えるだけで盤面の彩色と語の割当を自動で行うソフトウェアである。生成されるパズルの盤面は、日本語クロスワードパズルの慣例に加えて、周囲 3 マスが黒マスで囲まれたマスが存在しないという条件を満たしている。本実験では 32bit 版のクロスワードギバーを用いる。

クロスワードパズル自動生成に対する従来手法では、入力として彩色された盤面が必要であるため、辞書と盤面の大きさのみを入力とする本実験の適用は不可能である。

[実験 I] 提案手法とクロスワードギバーに対して、サイズと内容が異なる複数の辞書と、盤面の大きさを与えることで何らかのパズルが生成されるかを見る。盤面の大きさについては、提案手法には 1 辺の大きさnを与えて、クロスワードギバーには$n \times n$を与える。入力する辞書はサイズ 40 から 140 まで 20 刻みで 6 種類を用意した。また、1 辺の大きさnについては、6 から 9 までの 4 つを与える。提案手法では連結制約を課す場合と課さない場合に分けて実験を行う。連結制約を課す場合は目的関数 1 を用いて、連結制約を課さない場合は目的関数 3 を用いる。目的関数 3 のペナルティについては 1 と 2 を設定した。提案手法における計算時間については、nの大きさごとに 120 秒から 3,600 秒の範囲で設定し、計算時間内における最良解を出力する。提案手法に対する入力の組み合わせは、連結制約を課さない場合は、6 種類の辞書と 4 つのn、2 つのペナルティにより 48 通りである。連結制約

を課す場合は、6 種類の辞書と 4 つのnにより 24 通りである。クロスワードギバーに与える入力の組み合わせは、6 種類の辞書と 4 つのnにより 24 通りである。

[実験 II] 連結制約なし、目的関数 3、ペナルティ 1 のもとで、1 辺の大きさnに関わらず計算時間を一律 3,600 秒に設定した上で、パズルの生成を行う。3,600 秒以内で、最良解の更新が最後に行われた時間を記録し、次数nや辞書サイズとの関係を見る。入力の組み合わせは 6 種類の辞書と 4 つのnにより 24 通りである。

4.2 実験 I

実験の結果、提案手法では連結制約なしの場合の入力の組み合わせ 48 通りと、連結制約ありの場合の入力の組み合わせ 24 通りの全ての場合において、何らかのパズルを得ることができた。クロスワードギバーでは入力の組み合わせ 24 通りの全ての場合において、いかなるパズルも出力しなかった。よって、クロスワードギバーでは出力を得られない場合でも、提案手法を用いることで何らかの出力が得られることがわかった。また、提案手法では全ての場合において何らかの出力が得られたことから、定式化が間違っていないことがわかった。

クロスワードギバーから出力を得られなかった理由としては、入力された辞書に含まれる語を用いて、先に述べた盤面の条件を満たす解を発見できなかったためと考えられる。提案手法では、クロスワードギバーが持つ盤面生成の条件に従うことはせず、多少盤面の形が悪くてもパズルを出力することを優先するため、必ず何らかの出力が得られる。

出力の例を次頁の図 2、図 3、図 4 に示す。図 2 は北海道の地名 120 語から成る辞書を入力として、目的関数 3、ペナルティ 1、連結制約なしで生成したパズルである。図 3 は野菜や果物に関する 140 語から成る辞書を入力として、目的関数 3、ペナルティ 1、連結制約なしで生成したパズルである。図 4 は北海道の地名 120 語から成る辞書を入力として、目的関数 1、連結制約ありで生成したパズルである。

図 2 は 2 つの黒マスが隣接しているが、全ての白マスが連結している。一方で図 3 では、隣接した黒マスの対が 2 つあり、白マスも非連結である。図 3 は黒マスにより分断された 3 つの部分パズルが存在するため、その**連結成分数**は 3 である。図 2 と図 3 で示すパズルは、連結制約を課していないため、場合によっては連結成分数が 1 より大きいパズルも生成されてしまうこともあることがわかった。図 4 は連結制約ありで生成したパズルであり、全ての白マスが連結しているが、黒マスの連続が目立つ。連結制約なしのもとでは次数nが 7 で連結成分数 1 のパズルが一つも生成されなかったが、連結制約を課すことでこのようなパズルの出力を得ることもできる。

次頁の表 1 に辞書サイズ 120、目的関数 3、ペナルティ 1、連結制約なしで生成を行った際の詳細な結果を示す。表 1 を見ると、次数nが 6 のときは黒マス連続数 1、連結成分数 1 という質の良いパズルが生成されている。一方で次数nが大きくなるに伴い、黒マス連続数や連結成分数の増加が見られ、質の悪いパズルも生成されている。この結果については健

闘できたと考えられる。その理由として、本実験は入力される辞書サイズが小さいという不利な状況で行われたということが挙げられる。クロスワードギバーにてカナ辞書の場合は 5,000 語以上を含んだ辞書の使用を推奨していることからも、本実験で使用した辞書のサイズが非常に小さいということがわかる。小さい辞書を与えながらもこのような結果が得られたことから、健闘できたと考えられる。

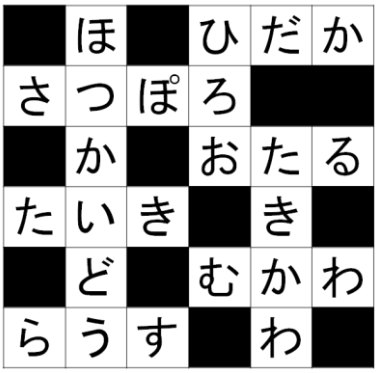


図 2 北海道パズル（連結制約なし）

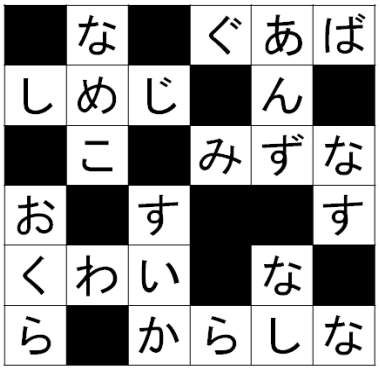


図 3 野菜・果物パズル（連結制約なし）

図 4 北海道パズル（連結制約あり）

表 1 辞書サイズ 120,目的関数 3,ペナルティ 1 のときの生成結果

n	6	7	8	9
計算上限時間	120	600	1800	3600
白マス数	24	34	42	54
黒マス数	12	15	22	27
黒マス連続数	1	4	0	5
連結成分数	1	2	6	5

4.3　実験 II

図 5 に生成実験 II を行った際の最良解を発見した時間のグラフを示す。縦軸は、最良解が発見された時間（秒）である。横軸は、次数nである。グラフ上の線の色によって、辞書サイズの違いを表す。実験 II では計算時間を 3,600 秒に設定しているため、3,600 秒以降に最良解の更新が行われる場合については調査しきれていない。しかしながら、クロスワードパズルの生成時間として 3,600 秒を超える時間を設定することは現実的ではないため、3,600 秒以内での最良解を発見した時間で判断する。図 5 を見ると、辞書サイズが 40 と 60 のとき、次数nが 6 から 8 までの間は短い時間で最良解の更新が終了している。辞書内の文字や単語長の分布などの影響もあるため一概には言えないが、辞書サイズが小さく、次数nも小さいときには短い時間で最良解の更新が止まることがわかった。また、辞書サイズ 80 を除いた全ての辞書において、次数nが 9 のときに最終更新時間が 3,600 秒に近付いていることから、提案手法が 3,600 秒で生成可能なパズルの限界は、次数nが 9 のパズルであるということもわかった。

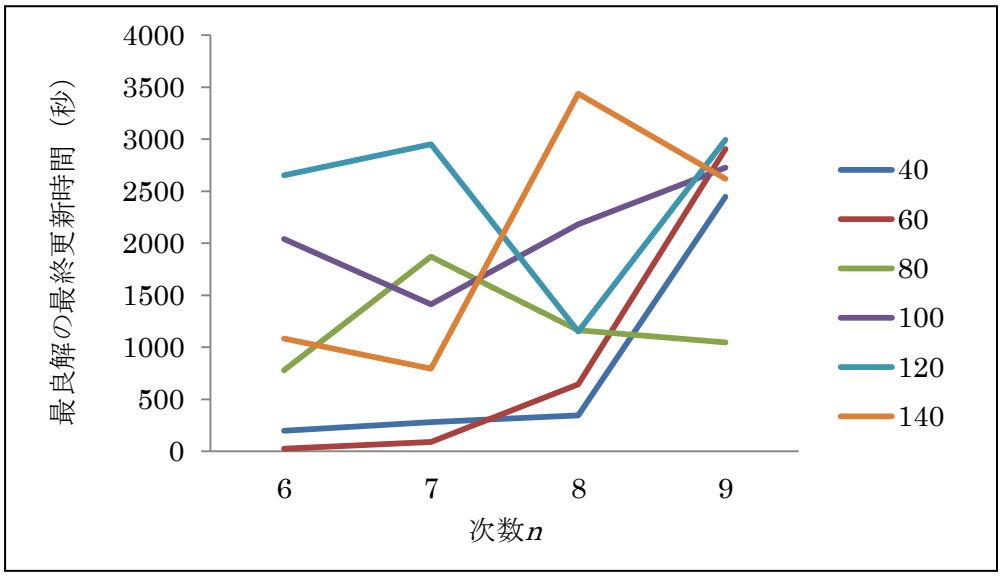


図 5 計算時間 3,600 秒以内での最良解の最終更新時間

5. 応用可能性

本章では、提案手法を用いて生成したパズルをいかにして応用するか、その可能性を 2 種類に分けて検討する。1 つ目として、提案手法を別のパズルの自動生成に利用できないか、その可能性を検討する。また、2 つ目として、提案手法を用いて生成したクロスワードパズルを観光振興に応用できないか、その可能性を検討する。

5.1　スケルトンパズル自動生成への応用

スケルトンパズルでは全てのマスが連結した盤面と、単語のリストが与えられる。図 6 にパズルの例を示す。リストの言葉を全て用いること、リスト中の言葉は 1 度しか使用できないこと、1 マスには 1 文字しか入らないこと、語を盤面の特定の場所に割り当てるときは上から下もしくは左から右に割り当てることというルールのもとで、リストに存在する全ての語を盤面に割り当てることができればパズルは完成である[10]。

提案手法を用いて、スケルトンパズルを生成することを考える。スケルトンパズルの盤面は全てのマスが連結しているため、連結制約を課した上で提案手法を用いることにより、スケルトンパズルの盤面を作ることが可能である。また、単語リストの生成については、提案手法の出力から単語を抜き出すことで生成が可能である。従って、形だけであればスケルトンパズルの生成が可能である。

しかしながら課題も存在する。パズルの性質上、解は唯一であることが望ましいが、提案手法を用いて生成されるスケルトンパズルがその性質を満たすとは限らない。解の唯一性を持たせるための制約を設定するなどの対応が必要である。

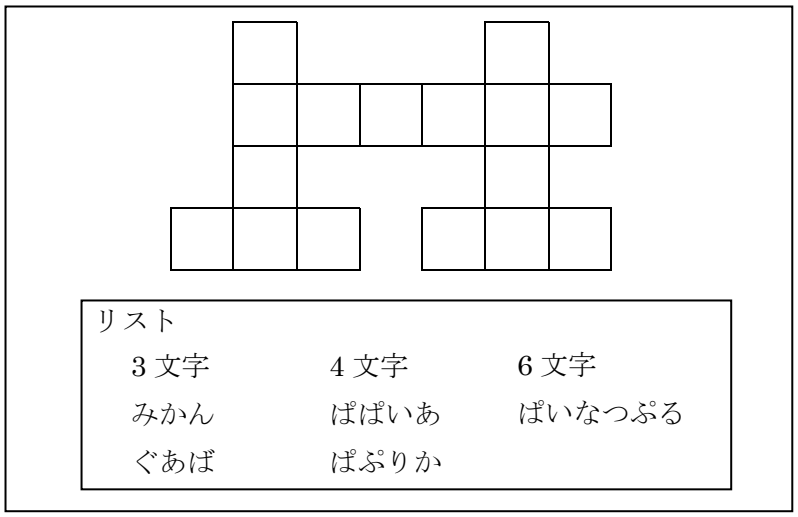


図 6 スケルトンパズルの例

5.2　観光振興への応用

・お土産としての利用

大量生産したパズルを 1 つにまとめて、地域観光のお土産として販売するという応用が考えられる。提案手法を用いて、ご当地単語を含む複数のクロスワードパズルを大量生産する。これらのパズルを、1 冊の本にまとめるなどしてお土産として販売する。観光客はそのお土産を購入し、観光旅行終了後に思い出を辿りながら解く。パズルを解く過程で再びその観光地に興味を持てば、リピーターになることが期待できる。

・クーポン券としての利用

完成したパズルを特定の店舗に持っていくことでクーポン券として利用できるという仕組みが考えられる。提案手法を用いて生成したパズルを観光客に配布する。観光客は移動時間等を利用してパズルに回答する。パズルが完成した暁には、特定の店舗に提示することでサービスが受けられるという仕組みである。観光地での消費を促す効果が期待できる。

・スタンプラリー型パズルとしての利用

スタンプラリーの要領で観光地を巡りながらパズルを解いていく仕組みが考えられる。盤面と解答は提案手法を用いて生成するが、カギにはヒント文を直接記述するのではなく、観光地の名前だけを記述する。観光客はカギに書いてある観光地を巡ることで、実際のヒント文を入手し、パズルを解いていく。パズルが完成した暁には記念品がもらえる等の特典を設定することで、回答への動機づけを行う。

この仕組みではカギに観光地の名前を記載するが、観光地の名前に加えてその観光地の説明文を載せることで、観光地紹介の役割をもたせることも可能である。

・観光地探索型パズルとしての利用

パズルのカギを工夫することで、観光地をくまなく探索しながらパズルを解いていく仕組みが考えられる。盤面と解答は提案手法を用いて生成するが、カギは観光地をくまなく探索しなければ答えられないほど難しいものを設定する。観光客は、カギを元に観光地をくまなく探索することで答えを探す。例えばカギとして「観光地 A の 2 階，銅像 B の隣にある展示物の名前」という文を設定する。このカギの答えは実際に観光地 A を訪れて銅像 B を探さなければわからないため、解くためには観光地 A を訪れた上で観光地内をくまなく探索する必要がある。こちらもパズルが完成した暁には記念品がもらえる等の特典を設定することで、回答への動機づけを行う。

この仕組みではパズルを完成させるために観光地をくまなく探索する必要があるため、学習と観光の両方を目的として訪れている修学旅行生等に配布することで、旅行の目的達成に高い効果が期待できる。

6. まとめ

本論文では、クロスワードパズル生成問題の新しい定式化を提案した。また、新しい定式化のもとで、辞書サイズが小さい場合でも必ず何らかの出力が得られることを確認した。従来の定式化では、何らかの目的によりパズルで使用できる語の数が限られる場合に、パズルを生成できないことがあったが、提案手法を用いることでパズルの生成が可能になった。

今後の課題として 3 つを挙げる。1 つ目は、提案手法で制約最適化に基づいている箇所について他の手法も試みることである。提案手法では制約最適化に基づく定式化を行うことで自動生成を実現したが、他のソルバーを用いることや遺伝的アルゴリズムなどの別の手法を用いることで、より良いパズルをより高速に自動生成できないか、その可能性を検討する必要がある。2 つ目は提案手法で良いパズルを生成するための条件として、どのような特徴を持つ辞書を入力すればよいかを調べることである。生成実験では 6 種類の辞書を与えたが、辞書やパズルのサイズによって連結成分数が 1 であるパズルが生成された一方、黒マス連続数や連結成分数が多いという質の悪いパズルも生成された。入力される辞書サイズが小さい場合、生成されるパズルに対する辞書の影響が大きくなると考えられるため、どのような特徴を持つ辞書を入力するべきかを調べる必要がある。3 つ目は辞書やカギを自動生成できるようにすることである。提案手法では彩色された盤面の入力をなくすことに成功したが、辞書についてはまだ何らかの方法で生成しなければならない。この辞書について、辞書に含む単語のテーマを与えるだけで自動的に単語を収集するといったエンジンを開発できれば、クロスワードパズルの自動生成がより簡単になると考えられる。また、カギについても自然言語処理や情報検索等の技術を用いて自動生成できるようになることが望まれる。

最後に付録として、提案手法を用いて生成したパズルを 1 つ掲載する。パズルは小樽商科大学に関する単語 120 語を用いて計算時間 120 秒、目的関数 3、ペナルティ 2 で生成したものである。カギについては筆者が作成したものを載せる。提案手法を用いることで付録に示すようなパズルを生成することが可能である。5 章で検討した観光振興への応用も含め、提案手法を用いて生成したパズルが様々な分野で活用されることを期待する。

謝辞

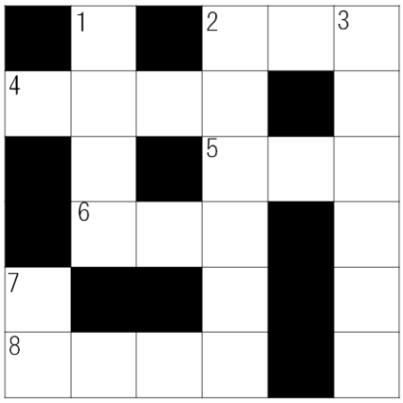
本論文の執筆に当たり、多くの方々にご協力いただきました。ご指導いただいた先生、学生論文賞で貴重なご意見をくださった先生方、学外での発表時に貴重なご意見をくださった方々には心より感謝を申し上げます。

参考文献

[1] 西勇樹, 小尻智子. 英語長文読解学習のためのクロスワード・パズルの自動生成手法. IEICE tech. rep., Education technology, 111(473):95-100, 2012.
[2] クロスワードパズル(幼児・小 1 用) | ぷりんときっず
http://print-kids.net/print/kokugo/crossword-puzzle/
(2016 年 12 月 21 日閲覧)
[3] hiroyuki endo : 英単語クロスワード TOEIC 600.
https://itunes.apple.com/jp/app/ying-dan-yukurosuwado-toeic/id831283001?mt=8 (App Store より)
(2016 年 12 月 21 日閲覧)
[4] Ichilisha LLC : 日本列島 全国クロスワード(脳トレパズル)の旅.
https://itunes.apple.com/jp/app/ri-ben-lie-dao-quan-guokurosuwado/id972424936?mt=8 (App Store より)
(2016 年 12 月 21 日閲覧)
[5] 加部通明, 生方俊典. クロスワードパズルの作成手法. 第 50 回情報処理学会全国大会講演論文集, pp157-158, 1995.
[6] Viet Ha Nguyen, 石川勉, 金杉友子. 制約充足的手法を応用したクロスワードパズルの解法. 第 56 回情報処理学会全国大会講演論文集, pp.314-315, 1998.
[7] クロスワード自動作成ソフト「クロスワードギバー」
http://katahiromz.web.fc2.com/xword/
(2016 年 12 月 21 日閲覧)
[8] 福富崇博. 日本語クロスワードパズルにおけるヒント文自動生成. 筑波大学第三学郡情報学類卒業研究論文, 2006.
[9] IBM ILOG CPLEX.
http://www-03.ibm.com/software/products/ja/ibmilogcple
(2016 年 12 月 21 日閲覧)
[10] スケルトンパズルの遊び方、ルール、解き方 | Web ニコリ
http://www.nikoli.co.jp/ja/puzzles/skeleton_puzzle/
(2016 年 12 月 23 日閲覧)

付録

小樽商科大学パズル



カギ

縦

1　3 号館 2 階、国際交流〇〇〇〇

2　入学式や卒業式を行う場所

3　先生になりたかったら〇〇〇〇〇〇科目を履修しよう

7　3 年次から所属する

横

2　以前は存在した小樽商科大学〇〇〇大学部

4　一般参加可能な〇〇〇〇講座

5　『若い詩人の肖像』著者、〇〇〇整

6　〇〇〇坂を登れば商大

8　緑丘祭で行われるコンテスト、男性版もある

解答

■	1ら	■	2た	ん	3き
4こ	う	か	い	■	よ
■	ん	■	5い	と	う
■	6じ	ご	く	■	し
7ぜ	■	■	か	■	よ
8み	す	こ	ん	■	く