ア ル ゴ リ ズ ム 80－3
（２００１．９．２５）

禁止領域を持つ格子スタイナー木問題の発見的解法DR

橋目 昭彦　高藤 大介　田岡 智志　渡邉 敏正

広島大学大学院 工学研究科 情報工学専攻
〒 739-8527 東広島市鏡山一丁目 4-1
(電話) 0824-24-7662 (渡邉), -7661(高藤), -7666 (田岡)
(ファクシミリ) 0824-22-7028
(電子メール) {hashime, takafuji, taoka, watanabe}@infonets.hiroshima-u.ac.jp

あらまし　格子スタイナー木問題とは，格子グラフ $H=(N,A)$，禁止領域集合族 $\mathcal{D}=\{D_i \subseteq N \mid 1 \leq i \leq j\}$ と指定点集合 P が与えられたとき，$H-\bigcup_{D_i\in\mathcal{D}} D_i=(V,E)$ において，指定点をすべて含み，辺重みの総和が最小となる格子スタイナー木 R_{ST} を求める問題である．但し，$H-\bigcup_{D_i\in\mathcal{D}} D_i$ は H から $\mathcal{D}$ の各点とそれに接続する辺を取り除いたグラフである．本稿では，$\mathcal{D}=\emptyset$ なる場合については，既存手法を組み合わせた改良手法を提案するとともに，既存手法で $\mathcal{D}\neq\emptyset$ なる場合を扱うための拡張法を述べる．一方，$\mathcal{D}\neq\emptyset$ なる場合については，まず高速解法 DD を提案し，領域（格子グラフである部分グラフ）分割して，次に，各領域に DD を適用する手法 DR を提案する．さらに，それらの手法の性能を実験的に評価する．

キーワード　格子スタイナー木, 禁止領域, 発見的解法

A heuristic algorithm DR for the rectilinear Steiner problem with rectangular obstacles

Akihiko Hashime,　Daisuke Takafuji,　Satoshi Taoka　and　Toshimasa Watanabe

Graduate School of Engineering, Hiroshima University
1-4-1, Kagamiyama, Higashi-Hiroshima, 739-8527 Japan
Phone : +81-824-24-7662 (Watanabe), -7661 (Takafuji), -7666 (Taoka)　Facsimile : +81-824-22-7028
E-mail : {hashime, takafuji, taoka, watanabe}@infonets.hiroshima-u.ac.jp

Abstract　The rectilinear Steiner tree problem with a family $\mathcal{D}$ of rectangular obstacles $D_i(1 \leq i \leq j)$ is defined as follows: given a rectilinear graph $H=(N,A)$, a set P of terminal points, and a family $\mathcal{D}$ of rectangular obstacles, find a minimum cost rectilinear Steiner tree connecting P in the graph $H-\bigcup_{D_i\in\mathcal{D}} D_i$ which is a graph obtained by removing from H all points in $\mathcal{D}$ and all edges incident to points in $\mathcal{D}$. For $\mathcal{D}=\emptyset$, we propose an improved algorithm by combining existing ones, and then describe how to extend it to be used in the case with $\mathcal{D}\neq\emptyset$. For $\mathcal{D}\neq\emptyset$, we first propose a fast heuristic algorithm DD . Also proposed is a heuristic algorithm DR based on partitioning H into some regions (rectilinear subgraphs), followed by applying DD to each region. Evaluating the performance of these algorithms through experimental results is also given.

key words　rectilinear Steiner trees,　rectangular obstacles,　heuristic algorithms

1 はじめに

格子スタイナー木問題とは次のように定義される：“格子グラフ $H=(N,A)$，禁止領域族 $\mathcal{D}=\{D_i \subseteq N \mid 1 \le i \le j\}$ と指定点集合 $P \subseteq N - \bigcup_{D_i \in \mathcal{D}} D_i$ が与えられたとき，$H - \bigcup_{D_i \in \mathcal{D}} D_i = (V,E)$ において，指定点をすべて含み，長さ（辺重み）の総和が最小となる木（格子スタイナー木と呼ぶ）$St(P)$ を求めよ”．但し，H における各 D_i の誘導部分グラフ $H[D_i]$ は連結であり，かつ $D_i \cap D_j = \emptyset (i \neq j)$ である．また，$H - \bigcup_{D_i \in \mathcal{D}} D_i$ は H から $\mathcal{D}$ の各点とそれに接続する辺を取り除いたグラフであり，特に断りのない限り $G=(V,E)$ と表す．特に必要がない限り，D_i と $H[D_i]$ を区別せず用いることとし，$H[D_i]$ も禁止領域と呼ぶ．

本問題は VLSI やプリント基板設計などに多くの応用を持つ．$\mathcal{D} \neq \emptyset$ の場合が応用上極めて重要であるが，今回の主題に関するサーベイでは $\mathcal{D} \neq \emptyset$ についての既存結果は見当たらなかったので，以下では $\mathcal{D}=\emptyset$ の場合の既存結果を簡単に紹介する．$|P| \ge 4$ ならば NP-完全問題である [2] ので，様々な近似解法が提案されている．なお，近似解法の性能評価の基準としては，performance ratio $PR = \max\{\frac{C'}{C} \mid C$ は最適解の重み総和, C' は近似解の重み総和 $\}$ を用いる．既存結果の中で次の 2 つが注目すべき解法である．ここで，入力グラフの頂点数，辺数，指定点数をそれぞれ $|N|$，$|A|$，$|P|$ と表記する．

BR： $PR=11/8$ なる $O(|P|^{5/2})$ の近似解法 [1]；
FK： $PR=3/2$ なる $O(|P|\log|P|)$ の近似解法 [3].

また，禁止領域が存在する格子グラフは，平面グラフで最大次数が 4 以下のグラフと考えることもできる．一般的なグラフに対するスタイナー木問題に対する既存解法では，次の 2 つに注目すべきある．

KM： $PR=2$ なる $O(|E|+|V|\log|V|)$ の近似解法 [6]；
BRG： $PR=16/9$ なる $O(|V|^5)$ の近似解法 [1].

本稿の目的は，各 $H[D_i]$ が矩形，格子グラフ（定義は次節参照）で辺重みが均一（ここではすべて 1 としている）の場合について，次の (1)〜(4) である．(1) $\mathcal{D}=\emptyset$ なる場合について、既存手法を組み合わせた改良手法の提案．(2) 既存近似解法 **BR**, **FK** をそれぞれ $\mathcal{D} \neq \emptyset$ なる場合への拡張手法の提案．(3)（応用上は大規模データを扱う必要があるので）高速な発見的解法 **DD**，領域分割に基づく発見的解法 **DR** の提案．(4) これらの提案解法を計算機上で実装し，実験的に性能を評価すること．

2 諸定義

xy-平面の第 1 象限とその中の整数座標を持つ点 $v=\langle x_v, y_v\rangle$ に着目する．2 組の非負整数対 (b_l,b_r), (h_b,h_t) が与えられたとき，各 $i(b_l \le i \le b_r)$ に対して，$\langle b_l,i\rangle$ と $\langle b_r,i\rangle$ を結び，x 軸に平行な n 本の線分，および各 $j(h_b \le j \le h_t)$ に対して，$\langle j,h_b\rangle$ と $\langle j,h_t\rangle$ を結び，y 軸に平行な n 本の線分，なる $2n$ 本の線分を考える．これらの交点を頂点とし，交点間を結ぶ線分を辺とするグラフを $(b_l,b_r)\times(h_b,h_t)$ の矩形格子グラフと呼び，特に $b_l=h_b=1$ かつ $b_r=h_t=n$ の場合を単に $n \times n$ の矩形格子グラフと呼ぶ．また，$n \times n$ の格子グラフが含まれる xy-平面の領域を $n \times n$ の格子領域と呼ぶ．H が $(b_l,b_r)\times(h_b,h_t)$ の矩形格子グラフで，Q を H の任意の点部分集合とする．このとき，H から Q を除去したグラフを $(b_l,b_r)\times(h_b,h_t)$ の格子グラフと（“矩形” を除いて）呼ぶ．$n \times n$ の格子グラフも同様に定義される．なお，本稿では H も各禁止領域 $H[D_i]$ も矩形格子グラフのみを扱うこととする．$H - \bigcup_{D_i \in \mathcal{D}} D_i$ は，一般には格子グラフである．

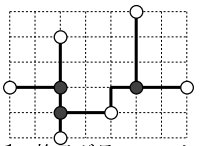

図 1: 格子グラフ H_1 とスタイナー木 T_1

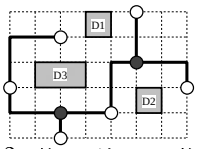

図 2: 禁止領域のある格子グラフ H_2 とスタイナー木 T_2

格子スタイナー木に含まれる頂点で，指定点ではなく，かつ次数が 3 以上の頂点をスタイナーポイントと呼ぶ．また，格子グラフにおいて，指定点集合 P の頂点をすべて含み，スタイナーポイントを持たない木の中で，重み総和最小の木を，P に対する最小スパニング木という．指定点集合 $S_i \subseteq P$ に対する格子スタイナー木 $St(S_i)$ の辺重み総和を $w(St(S_i))$ と表す．

格子グラフ H の任意の頂点集合 $N' \subseteq N$ に対し，下記のように表記する．

$$
\begin{aligned}
max_x(N') &= \max\{x_v \mid v=\langle x_v,y_v\rangle \in N'\},\\
min_x(N') &= \min\{x_v \mid v=\langle x_v,y_v\rangle \in N'\},\\
max_y(N') &= \max\{y_v \mid v=\langle x_v,y_v\rangle \in N'\},\\
min_y(N') &= \min\{y_v \mid v=\langle x_v,y_v\rangle \in N'\},\\
mid_x(N') &= \lceil (max_x(N')+min_x(N'))/2 \rceil,\\
mid_y(N') &= \lceil (max_y(N')+min_y(N'))/2 \rceil
\end{aligned}
$$

以下この節では，G は（$H - \bigcup_{D_i \in \mathcal{D}} D_i$ とは限らず）一般的なグラフの意味で用いる．頂点集合 V，辺集合 E，辺重み関数 $w_G : E \to Z^+$（非負整数）からなる辺重み付きグラフを $G=(V,E,w_G)$ と表す．逆にグラフ G の点集合，辺集合をそれぞれ $V(G)$，$E(G)$ と表す．なお，すべての辺重みが 1 の場合は w_G を省略する場合もある．G における $V' \subseteq V$ の誘導部分グラフを $G[V']$ と記す．u と v を両端点とする辺を (u,v) と表す．v_0, v_n 間の頂点と辺の順序列 $v_0,\ e_1,\ \ldots,\ e_{n-1},\ v_n$ をパスと呼ぶ．但し，$v_i \in V\ (0 \le i \le n)$, $e_i=(v_{i-1},v_i) \in E\ (1 \le i \le n-1)$ である．v_0，v_n

間のパスで辺重み総和が最小のものを（v_0，v_n 間の）最短パスという．任意の 2 頂点 $u, v \in V$ に対して，$d_{min}(u, v; G)$ を 2 点間の最短パスの重み総和とする．G が明らかな場合，$d_{min}(u, v; G)$ を単に $d_{min}(u, v)$ と記すこともある．

指定点集合を $P \subseteq V$ と表す．各 $p \in P$ に対して p の近傍 $N(p) \subseteq V$ を次のように定義する：$N(p) = \{v \in V \mid d_{min}(v, p) \leq d_{min}(v, p') \text{ for any } p' \in P - \{v, p\}\}$．但し，指定点対 $q_1, q_2 \in P$ に対し，$v' \in N(q_1) \cap N(q_2)$ なる頂点 $v' \in V$ が存在するならば，任意に $q_i \in \{q_1, q_2\}$ を選び，$N(q_i) = N(q_i) - \{v'\}$ とする．以上により頂点集合の族 $\mathcal{N} = \{N(p) \subseteq V \mid p \in P\}$ を定義すると，以下の (1)，(2) がともに成り立つ：(1) 任意の指定点 $q_1, q_2 \in P$ に対し，$N(q_1) \cap N(q_2) = \emptyset$；(2) $V = \bigcup_{N(p) \in \mathcal{N}} N(p)$．

点集合 P とグラフ $G = (V, E, w_G)$ の辺縮約グラフ $\overline{G_P} = (P, F, w_P)$ を次のように定義する：$F = \{(s, t) \mid s, t \in P$ かつ $u \in N(s), v \in N(t)$ なる $(u, v) \in E$ が存在する $\}$．各 $e = (s, t) \in F$ に対し，$w_P(e) = \min\{d_{min}(s, u; G) + w_G(e') + d_{min}(t, v; G) \mid u \in N(s), v \in N(t)$ なる $e' = (u, v) \in E\}$．

3 $\mathcal{D} = \emptyset$ なる場合

本節では，$\mathcal{D} = \emptyset$ なる場合の近似解法 **BR**[1]，**FK**[3] を概説し，これらを利用した改良解法を示す．**BR**，**FK** では，入力は指定点集合 $P \subseteq N$ と矩形格子グラフ $H = (N, A)$ （$G = (V, E)$ は H と同一），出力は格子スタイナー木 $St(P)$ である．

3.1 $PR = 11/8$ なる近似解法 BR[1]

$PR = 11/8$ なる近似解法 **BR** の概要を述べる．まず，P と H の辺縮約グラフ $\overline{H_P}$ を構成し，$\overline{H_P}$ の最小スパニング木を求める．この最小スパニング木の各辺を H の該当するパスに置き換えることにより H の格子スタイナー木を求める．ここで求まった格子スタイナー木について，$PR = 3/2$ となることが [4] で示されている．次に，$|P| = 3$ なる指定点集合 $P \subseteq N$ と $\mathcal{D} = \emptyset$ なる格子グラフ $H = (N, A)$ に対する格子スタイナー木問題では，多項式時間で厳密解を求めることができるので，$|S_i| = 3$ なるすべての集合 $S_i \subseteq P$ に対し，格子スタイナー木を求める．最後に，現在求まっている格子スタイナー木の辺と入れ換えることにより，解精度を向上させる．

なお，**Phase 2** で扱う頂点集合族 $\mathcal{S} = \{S_i \subseteq P \mid |S_i| = 3\}$ は，次をみたす：任意の $S_i \in \mathcal{S}$ に対し，$min_x(S_i) \leq x_p \leq max_x(S_i)$, $min_y(S_i) \leq y_p \leq max_y(S_i)$ なる $p = \langle x_p, y_p \rangle \in P - S_i$ は存在しない．

以下に **BR** の詳細を述べる．

近似解法 BR

Phase 1 /* $PR = 3/2$ なる格子スタイナー木の構成 */

(1) P と H の辺縮約グラフ $\overline{H_P} = (P, F, w_P)$ を構成し，$\overline{H_P}$ の最小スパニング木 T_1 を求める．

(2) 各辺 $e \in T_1$ を，H での該当するパス $Q(e)$ に置き換え，格子スタイナー木 $St(P)$ を求める．

Phase 2 スタック Y を $Y \leftarrow \emptyset$ とする．各 $e \in T_1$ に対し，$w_t(e) \leftarrow w_P(e)$ とする．

(1) 任意の $S_i \in \mathcal{S}$ に対し，以下 (i)–(iii) を行う．/* $S_i = \{p_1, p_2, p_3\}$ とする．*/

(i) T_1 での p_k（それぞれ p_j）を含まない p_j, p_m 間（p_k, p_m 間）のパス Q_j（Q_k）を求める．但し，$\{j, k, m\} = \{1, 2, 3\}$ である．

(ii) 各 $x \in \{j, k\}$ について $w_t(e_x) = \max\{w_t(e) \mid e \in E(Q_x)\}$ なる辺 $e_x \in E(T_1)$ に着目し，$gain(S_i) \leftarrow w_t(e_j) + w_t(e_k) - w(St(S_i))$ とする．但し，$St(S_i)$ は関数 **Local_for_three**(S_i, H) により得られた格子スタイナー木である．

(iii) もし，$gain(S_i) > 0$ ならば，各 $x \in \{j, k\}$ に対し，$w_t(e_x) \leftarrow w_t(e_x) - gain(S_i)$ とし，$\{e_j, e_k, S_i\}$ をスタック Y に挿入する．

(2) $Y = \emptyset$ でない限り，以下 (i), (ii) を行う．

(i) スタック Y から $\{e_1, e_2, S_i\}$ を取り出す．

(ii) $e_1 \in E(T_1)$ 且つ $e_2 \in E(T_1)$ ならば，$E(T_1) \leftarrow E(T_1) - \{e_1, e_2\}$, $E(St(P)) \leftarrow E(St(P)) \cup E(St(S_i)) - \{E(Q(e_1)), E(Q(e_2))\}$ とする．

以下に関数 **Local_for_Three()** の詳細を記す．なお，入力は $S = \{u_1, u_2, u_3\} \subseteq N$ と $H = (N, A)$ であり，出力は格子スタイナー木 $St(S)$ である．

関数 **Local_for_Three**(S, H)

step 1 $v_s = \langle mid_x(S), mid_y(S) \rangle \in N$ とする．v_s と各 $u_i \in S$ $(1 \leq i \leq 3)$ に対し，v_s，u_i 間の最短パス Q_i を求める．

step 2 $E(St(S)) \leftarrow \bigcup_{1 \leq i \leq 3} E(Q_i)$．

3.2 $PR = 3/2$ なる近似解法 FK[3]

$PR = 3/2$ 近似解法 FK[3] の概要を述べる．まずは，P と H の辺縮約グラフを $\overline{H_P} = (P, F, w_P)$ とし，$\overline{H_P}$ での最小スパニング木 T_1 の辺重みによって，P の各点に操作順を割り当てる．指定点集合 P からこの順番に 3 点を選び，その 3 点に関する最小格子スタイナー木を求め，それを $St(P_1)$ とする．そして，以下では、次の操作をできる限り繰り返す：『$St(P_i)$ に含まれない指定点で順番の早い 1 点を選び，$St(P_i)$ に含まれる 2 点と合わせた 3 点に関する最小格子スタイナー木を求め，$St(P_i)$ に加え，格子スタイナー木 $St(P_{i+1})$ を得る．』最終的に、格子スタイナー木 $St(P)$ が構成されることは明らかである．

3.3 提案手法

本節では，**BR** および **FK** の改良解法として，**FBRK**, **FKn**, **BRn** を提案する．

FBRK は，**FK** の実行後に得られた格子スタイナー木のすべてのスタイナーポイントを指定点集合 P に追加して P' とおき，P を P' に置き換えて入力として **BR** を実行し，格子スタイナー木を得る．さらに，得られた格子スタイナー木のすべてのスタイナーポイントを指定点集合 P' に追加して P'' とおき，P' を P'' に置き換えて入力とし，もう一度 **FK** を実行することで格子スタイナー木を得る．

BRn（それぞれ，**FKn**）は，**BR**（**FK**）を n 回反復する解法であるが，**BR**（**FK**）を $i(1 \leq i \leq n-1)$ 回反復後に得られる格子スタイナー木のすべてのスタイナーポイントを i 回目開始前の指定点集合 P_i に追加して P_{i+1} とし，P_{i+1} を $(i+1)$ 回目の指定点集合として **BR**（**FK**）を実行するものである．

4 $\mathcal{D} \neq \emptyset$ なる場合

本節では，$\mathcal{D} \neq \emptyset$ なる場合の近似解法 **BRG** と提案解法を述べる．4.3 節（それぞれ，4.4）では，既存手法 **BR**（**FK**）を $\mathcal{D} \neq \emptyset$ なる場合への拡張手法を説明する．4.5 節，4.6 節では，高速化を目指した提案解法を説明する．

そのためにいくつか定義と準備をしておく．なお，ここでも $H = (N, A)$ は格子グラフ，$G = (V, E)$ は $H - \bigcup_{D_i \in \mathcal{D}} D_i$ を表すものとする．

4.1 禁止領域と track graph

4.1.1 禁止領域の隣接性

H を禁止領域族 $\mathcal{D}$ を持つ格子グラフとする．$D_i, D_j \in \mathcal{D}(i \neq j)$ に対して，$x \in D_i$ かつ $y \in D_j$ なる辺 $(x, y) \in A$ が H に存在するとき，D_i と D_j は隣接する（あるいは隣接対である）ということにする．もしそのような点対がないときには，D_i と D_j は非隣接である（または非隣接対である）という．このとき，$D_i \cap D_j = \emptyset$ であるが，G を考えるときには $D_i \cup D_j$ を 1 つの禁止領域として扱ってよい．$D_i = D_1, D_2, \cdots, D_k = D_j (k \geq 2)$ なる禁止領域の列で，各 $t = 2, \cdots, k$ について「D_{t-1} と D_t は隣接対である」をみたすものがあるとき，D_i と D_j は連結であるといい，逆に，この様な列が存在しないとき，D_i と D_j は非連結であるという．すると，この連結性より $\mathcal{D}$ は $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_s (s \geq 1)$ と分割できる．ここで，各 $\mathcal{D}_k$ の任意の 2 つの禁止領域は連結であり，かつ $k \neq k'$ ならば $\mathcal{D}_k$ 内の禁止領域と $\mathcal{D}_{k'}$ の禁止領域で連結なものは存在しない．$\delta(\mathcal{D}) = \max\{|\mathcal{D}_1|, \cdots, |\mathcal{D}_s|\}$ とおき，これを $\mathcal{D}$ の次数とよぶ．各 $\mathcal{D}_k$ について，それに属するすべての禁止領域の和集合を新しく 1 つの禁止領域と考えることにより，$\mathcal{D}$ から新しく禁止領域族 $\hat{\mathcal{D}}$ が定められる．但し，$\hat{D}_i, \hat{D}_j \in \hat{\mathcal{D}}(i \neq j)$ に対しては $\hat{D}_i$ と $\hat{D}_j$ は非連結である．$\hat{\mathcal{D}}$ を非連結な禁止領域族とよぶ．

本稿では $\delta(\mathcal{D}) \leq 2$ なる禁止領域族 $\mathcal{D}$ を考える．したがって，任意の $D_i \in \mathcal{D}$ に対して，$\hat{D}_i = D_i \in \hat{\mathcal{D}}$ であるか，またはある $D_j \in \mathcal{D}(j \neq i)$ が存在して，$\hat{D}_{ij} = D_i \cup D_j \in \hat{\mathcal{D}}$ であるか（図 3 参照），いずれかである．

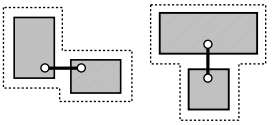

図 3: 隣接する禁止領域対とその近傍

4.1.2 近傍と track graph

$\delta(\mathcal{D}) \leq 2$ とし，各 $\hat{\mathcal{D}}_k \in \hat{\mathcal{D}}$ に対して，その近傍 $NB(\hat{\mathcal{D}}_k)$ を以下のような点集合 $V(NB(\hat{\mathcal{D}}_k))$ による G での誘導部分グラフとして定義する:$V(NB(\hat{\mathcal{D}}_k)) = \{v \in V \mid (u, v) \in A$ なる $u \in \hat{\mathcal{D}}_k$が存在する $\}$．$\delta(\mathcal{D}) \leq 2$ であることから，$V(NB(\hat{\mathcal{D}}_k))$ は一般的には 1 つのサイクルである（図 4 参照）．但し，$\hat{\mathcal{D}}_k$ が H の最外側の点を含むときは切断された形の 1 本以上のパスとなる．$NB(\hat{\mathcal{D}}_k)$ と $NB(\hat{\mathcal{D}}_{k'})(k' \neq k)$ は点や辺を共有することがある．

次に，P と H と $\mathcal{D}$ に関する track graph を説明する．G から次の (1),(2) をみたす辺 (u, v) をすべて除去する：(1) (u, v) はどの近傍にも含まれない；(2) $u = \langle u_x, u_y \rangle$，$v = \langle v_x, v_y \rangle$ とするとき，x 座標が u_x または v_x である指定点も，y 座標が u_y または v_y である指定点も，いずれも存在しない．この結果，得られるグラフ $TR_{P,H,\mathcal{D}} = (V_{P,H,\mathcal{D}}, E_{P,H,\mathcal{D}})$ を（P と H と $\mathcal{D}$ に関する）track graph とよぶ．track graph はその定義からわかるように，$V_{P,H,\mathcal{D}} = V(= N - \bigcup_{D_i \in \mathcal{D}} D_i)$ であり，$E_{P,H,\mathcal{D}} \in E$ は近傍かまたは指定点を通り，x 軸または y 軸に平行な線分に含まれるような G の辺からなる．

track graph は [5] において定義されているが，[5] では頂点集合しか定義されていないため，本稿では辺集合もあわせて定義したものを track graph とする．

対象とするグラフを track graph に置き換えることにより，グラフのサイズが小さくなり，大規模なグラフを扱うことができると考えられる．

以下では P や $\mathcal{D}$ が固定された場合には，これらを省略して，$TR_{P,H,\mathcal{D}} = (V_{P,H,\mathcal{D}}, E_{P,H,\mathcal{D}})$ を $TR(H) = (V_{TR}, E_{TR})$ と表すこともある．

4.2 既存近似解法 BRG[1]

BRG は一般的な辺重み付きグラフ $G = (V, E, w)$

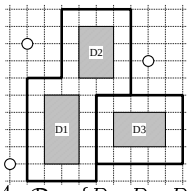

図 4: $\mathcal{D} = \{D_1, D_2, D_3\}$ と $P = \{p_1, p_2, p_3\}$ を持つ格子グラフの例．$D_1 \cup D_2$，および D_3 各々の近傍を太実線で示す．

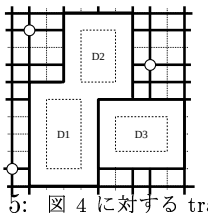

図 5: 図 4 に対する track graph

と指定点集合 $P \subseteq V$ に対するスタイナー木問題の解法で，$PR = 16/9$ である．本稿で扱う格子スタイナー木問題において，与えられた辺重み付きグラフは最大次数が 4 以下である平面グラフであるので，**BRG** が適用できる．

BRG は，**BR** から拡張されたものであり，**BR** と異なる点は，(1) **Phase 2** での頂点集合族 $\mathcal{S} = \{S \subseteq P \mid |S| = 3\}$ の選び方に条件がないことと；(2) 関数 **Local_for_Three()** の違い，である．その関数 **Local_for_Three()** の詳細を示す．入力は，$|S| = 3$ なる $S = \{u_1, u_2, u_3\} \subseteq P$ と辺重み付きグラフ $G = (V, E, w)$ であり，出力は S に含まれる 3 点を連結するスタイナー木 T である．なお，辺重みは 1 であるので w は省略する．

関数 **Local_for_Three**(S, G)

step 1 $V' = N(u_1) \cup N(u_2) \cup N(u_3)$ とし，任意の $v \in V' - S$ において，以下を行う．
- (1) 各 i $(1 \le i \le 3)$ に対し，G での v, u_i 間の最短パス Q_i を求める．但し，$1 \le i, j \le 3$ なる任意の i, j $(i \ne j)$ に対し，$V(Q_i) \cap V(Q_j) = \{v\}$ を満たす．
- (2) $E(T(S; v)) \leftarrow E(Q_1) \cup E(Q_2) \cup E(Q_3)$.

step 2 各 $v \in V' - S$ におけるスタイナー木 $T(S; v)$ の中で，重み総和最小なものを $T(S)$ とする．

4.3 BR の拡張手法

BR の 4 つの拡張版 **BR′**，**BR**$_r$，**BR**$_p$，**BR**$_{r+p}$ を説明する．まず，1 つ目の拡張点は対象となるグラフ $H = (N, A)$ を P と $\mathcal{D}$ と H の track graph $TR(H) = (V_{TR}, E_{TR}, w_{TR})$ とすることである．このため，Phase 1 では，重み付きグラフ $\overline{H_P}$ を $\overline{TR_P}$ と置き換え，$\overline{TR_P}$ で最小スパニング木 T_1' を求めることとする．track graph の定義より，$|V_{TR}| \le |V|$ 且つ $|E_{TR}| \le |A|$ となる．このため，対象となる格子グラフ $H = (N, A)$ を $G = (V, E)$ に置き換えるよりも track graph に置き換えた方がグラフのサイズが小さくなり，逆に，より大規模なグラフを扱うことができると考えられる．しかし，T_1' と（$\overline{H_P}$ の最小スパニング木）T_1 の辺重み総和が等しいことは [5] で示されているが，（P と $\mathcal{D}$ を固定して）H における格子スタイナー木と，TR における格子スタイナー木の辺重み総和が必ずしも等しくはならない．

2 つ目の拡張点は，$\mathcal{D} \ne \emptyset$ の場合を扱うための関数 **Local_for_Three()** の拡張である．**BR** での関数 **Local_for_Three()** をそれぞれ，**BR′** は [A] に，**BR**$_r$ は [B] に，**BR**$_p$ は [C] に，**BR**$_{r+p}$ は [D] に置き換えたものである．この関数の入力は $S = \{u_1, u_2, u_3\} \subseteq V_{TR}$ と $TR(H) = (V_{TR}, E_{TR}, w_{TR})$ であり，出力は格子スタイナー木 $St(S)$ である．

[A] 関数 **Local_for_Three**$(S, TR(H))$

step 1 $V' = \{v \in V_{TR} \mid min_x(S) \le x_v \le max_x(S)$ and $min_y(S) \le y_v \le max_y(S)$, where $v = \langle x_v, y_v \rangle\}$ とし，任意の $v \in V' - S$ に対して，以下を行う．
- (1) 各 i $(1 \le i \le 3)$ に対し，$TR(H)$ での v, u_i 間の最短パス Q_i を求める．但し，$1 \le i, j \le 3$ なる任意の i, j $(i \ne j)$ に対し，$V(Q_i) \cap V(Q_j) = \{v\}$ を満たす．
- (2) $E(St(S; v)) \leftarrow E(Q_1) \cup E(Q_2) \cup E(Q_3)$.

step 2 各 $v \in V' - S$ についての格子スタイナー木 $St(S; v)$ のうちで，重み総和が最小なものを $St(S)$ とする．

[B] 関数 **Local_for_Three**$(S, TR(H))$

step 1 $S' = \{u_j, u_k\} \subset S$ を任意に選ぶ．

step 2 $B_x \leftarrow min_x(S')$, $U_x \leftarrow max_x(S')$, $B_y \leftarrow min_y(S')$, $U_y \leftarrow max_y(S')$ とする．

step 3 $TR(H)[V']$ が非連結である限り，$B_x \leftarrow B_x - 1$, $U_x \leftarrow U_x + 1$, $B_y \leftarrow B_y - 1$, $U_y \leftarrow U_x + 1$ と順に繰り返す．但し，$V' = \{v = \langle x_v, y_v \rangle \in V_{TR} \mid B_x \le x_v \le U_x$ and $B_y \le y_v \le U_y\}$.

step 4 $TR(H)$ で $\{u_m\} = S - S'$ なる u_m を始点とする最短パス木を求め，$d_{min}(v_s, u_m) = \min\{d_{min}(v, u_m) \mid v \in V'\}$ を満たす $v_s \in V'$ を求める．

step 5 step 4 で求めた $v_s \in V'$ と各 i $(1 \le i \le 3)$ に対し，$TR(H)$ で v_s, u_i 間の最短パスを求める．但し，$1 \le i, j \le 3$ なる任意の i, j $(i \ne j)$ に対し，$V(Q_i) \cap V(Q_j) = \{v_s\}$ を満たす．

step 6 $E(St(S)) \leftarrow E(Q_1) \cup E(Q_2) \cup E(Q_3)$.

[C] 関数 **Local_for_Three**$(S, TR(H))$

step 1 S から任意に 2 点を選び，u_j, u_k $(j \ne k)$ とする．

step 2 $TR(H)$ で u_j, u_k 間の最短パス Q_1 を求める．

step 3 $\{u_m\} = S - \{u_j, u_k\}$ なる u_m に対し，u_m を始点とする $TR(H)$ での最短パス木を求め，$d_{min}(v_s, u_m) = \min\{d_{min}(v, u_m) \mid v \in V(Q_1)\}$ を満たす $v_s \in V(Q_1)$ を求める．

step 4 v_s, u_m 間の $TR(H)$ での最短パスを Q_2 とすると，$E(St(S)) \leftarrow E(Q_1) \cup E(Q_2)$.

[D] 関数 **Local_for_Three**$(S, TR(H))$

step 1 提案解法 $\mathbf{BR}_r$ により $St(S)_r$ を求める.

step 2 提案解法 $\mathbf{BR}_p$ により $St(S)_p$ を求める.

step 3 $St(S)_r$ と $St(S)_p$ のうち，重み総和の小さいものを $St(S)$ とする.

4.4 FK の拡張

FK の 4 つの拡張版 $\mathbf{FK}'$, $\mathbf{FK}_r$, $\mathbf{FK}_p$, $\mathbf{FK}_{r+p}$ は，**BR**4 種類の拡張解法と同じく，(1) 対象となるグラフを与えられた格子グラフから track graph への置き換え，(2) 関数 **Local_for_Three()** をそれぞれ，$\mathbf{FK}'$ は 4.3 節 [A] への，$\mathbf{FK}_r$ は 4.3 節 [B] への，$\mathbf{FK}_p$ は 4.3 節 [C] への，$\mathbf{FK}_{r+p}$ は 4.3 節 [D] への置き換え，により得られる.

4.5 提案手法 DD

BR を $\mathcal{D} \neq \emptyset$ なる場合に適用できるよう拡張した解法 **DD** について説明する．**BR** から (1) 対象となるグラフを格子グラフから track graph に置き換えたこと，および (2) Phase 2 を変更したものである.

ここでは Phase 2 の概要を述べる．P と $\mathcal{D}$ と H の track graph $TR(H) = (V_{TR}, E_{TR}, w_{TR})$ に着目する．まず，頂点 v_P（求め方は後述）をスタイナーポイントとする点集合 $P'(v_P) \subseteq P$ についての格子スタイナー木を求め，これを既存の格子スタイナー木 $St(P)$ に加える．但し，$P'(v_P) \subseteq P$ は v_P からの最短パス長が r 以下となる指定点の集合である．本稿では，$r = \lceil (\max_x(V_{TR}) - \min_x(V_{TR}))/2 \rceil$ とする．$(V_{TR}, E(St(P)))$ にサイクルが存在する限り，$St(P)$ から辺を除去する．これにより得られた格子スタイナー木を $St'(P)$ とすると，$w(St'(P)) < w(St(P))$ ならば，$St'(P)$ を格子スタイナー木 $St(P)$ として出力し，そうでなければ，$St(P)$ を出力する.

v_P の選び方を述べる．$v \notin P$ なる点 $v \in V_{TR}$ の中で，v からの最短パス長が r 以下となる指定点集合を $P'(v) \subseteq P$ とすると，$|P'(v)|$ が最大となる頂点を選ぶ．そのような頂点が複数存在する場合は，$center_P = \langle mid_x(P), mid_y(P) \rangle$ からの最短パス長が最小となるものを選ぶ.

以下，Phase 2 の詳細を記す.

Phase 2 $T' \leftarrow St(P)$ として，以下を行う.

(1) 各 $v \in V_{TR}$ に対し，$DIST(v) \leftarrow 0$ として，$DIST(v)$ を次のように計算する：各指定点 $p \in P$ において，TR での指定点 p からの最短パス長が r 以下となるすべての頂点 $v \in V_{TR}$ に対し，$DIST(v) \leftarrow DIST(v) + 1$.
$r = \lceil (\max_x(V_{TR}) - \min_x(V_{TR}))/2 \rceil$.

(2) $DIST(v) = \max\{DIST(v') \mid v' \in V_{TR}\}$ となる頂点 $v \in V_{TR}$ の中で，$center_P = \langle mid_x(P), mid_y(P) \rangle$ からの最短パス長が最小となるものを v_P とする.

(3) TR での v_P からの最短パス長が r 以下となる指定点集合を $P' \subseteq P$ とおく．関数 **Find_SPT** (P', TR, v_P) を行うことにより，v_P を根とする最短パス木 DT（但し，$E(DT) \subseteq E_{TR}$）を求める．但し，$TR_1 = (V_{TR}, DT)$ とすると，$P' \subseteq V_{TR}$ であり，TR_1 において各 $p' \in P$ は連結である.

(4) $add \leftarrow 0$ とし，各 $e \in DT$ において，$e \notin E(St(P))$ ならば，$E(St(P)) \leftarrow E(St(P)) \cup \{e\}$ 且つ $add \leftarrow add + w_{TR}(e)$ とする.

(5) $TR'' = (V_{TR}, E(St(P)))$ にサイクルが存在すれば，$E(St(P))$ からサイクル上の辺をサイクルが存在しなくなるまで除去する．また，$p \notin P$ なる葉が存在する限り，その葉に接続する辺 $e' \in E(St(P))$ を除去する.

(6) (5) で除去された辺の重み総和を del とする．$gain = del - add < 0$ ならば $St(P) \leftarrow T'$.

以下に，関数 **Find_SPT()** の詳細を記す．入力は指定点集合 $P' \subseteq V_{TR}$，track graph $TR = (V_{TR}, E_{TR}, w_{TR})$，頂点 $s \notin P'$ であり，出力は s を根とする最短パス木 DT である．TR の辺重みが 1 であることより，始点 s と任意の頂点 u $(n \neq s)$ における BFS 木の s, u 間のパスは，TR での最短パスとなる.

関数 **Find_SPT**(P', TR, s)

step 1 s を始点とし，幅優先探索により BFS 木 DT(但し, $E(T) \subseteq E_{TR}$) を求める．但し，任意の点 $v = \langle x_v, y_v \rangle \in V_{TR}$ からの幅優先探索での前進操作は，$z_1 = \langle x_v + 1, y_v \rangle$, $z_2 = \langle x_v - 1, y_v \rangle$, $z_3 = \langle x_v, y_v + 1 \rangle$, $z_4 = \langle x_v, y_v - 1 \rangle$ とすると，i の順に行う.
/* $(v, z') \in E_{TR}$ なる $z' \in V_{TR}$ は $z' \in \{z_i \mid 1 \leq i \leq 4\}$ である．*/

step 2 DT において，$w \notin P'$ なる葉 $w \in V_{TR}$ が存在する限り，以下を繰り返す：
各 $(w, w') \in E'$ に対し，$E' \leftarrow E' - \{(w, w')\}$.

4.6 提案解法 DR

提案解法 **DR** は，**BR** から，(1) 対象となるグラフを格子グラフから track graph に置き換え，(2) Phase 2 を変更したものである．ここでは，Phase 2 の概要を述べる.

DR では，格子領域を再帰的に 4 分割し，各分割領域ごとに 4.5 節の提案解法 **DD** を用いる.

以下，提案解法 **DR** の Phase 2 の詳細を記す.

Phase 2 $Div(TR(H)) \leftarrow 0$ とし，以下を行う．

(1) 関数 **Local_Solution**$(P, St(P), TR(H))$ を行うことにより，$St(P)$ を更新する．

(2) $St(P)$ にサイクルが存在する限り，サイクル上の辺を除去する．また，$v \notin P$ なる葉が存在する限り，その葉に接続する辺を除去する．

4.6.1 関数 Local_Solution()

関数 **Local_Solution()** は track graph $TR = (V_{TR}, E_{TR}, w_{TR})$，指定点集合 $P \subseteq V_{TR}$，格子スタイナー木 $St(P)$ が与えられたとき，$St(P)$ を重み総和のより小さい格子スタイナー木に変更する．与えられた TR のサイズが十分小さい場合は，TR で格子スタイナー木を求める．そうでない場合は，関数 **Div_Reg()** により TR を 4 分割し，新たに生成された track graph それぞれに対し，関数 **Local_Solution()** を行う．ここで track graph のサイズが十分に小さいかどうかの判定として，track graph TR が生成されるまでに関数 **Div_Reg()** が実行された回数 $Div(TR)$ を用いる．予め与えられた num_d に対し，$Div(TR) = num_d$ ならば，この TR のサイズは十分小さいものとする．

以下に，関数 **Local_Solution()** の詳細を記す.

関数 **Local_Solution**$(P, St(P), TR)$

step 1 $Div(TR) < num_d$ ならば，関数 **Div_Reg** (TR) を行い，$V_{TR} = \bigcup_{1 \le i \le 4} V^i_{TR}$ なる族 $\mathcal{TR} = \{TR^i = (V^i_{TR}, E^i_{TR}, w^i_{TR},) \mid 1 \le i \le 4\}$ を求める．$Div(TR) = num_d$ ならば，step 3 へ.

step 2 各 $TR^i \in \mathcal{TR}$ $(1 \le i \le 4)$ に対し，$Div(TR^i) \leftarrow Div(TR^i) + 1$ とする．$P^i = P \cap V^i_{TR}$, $E(St(P^i)) = E(St(P)) \cap E^i_{TR}$ とする．

step 3 関数 **Local_Solution**$(P^i, St(P^i), TR^i)$ を行う.

step 4 track graph $TR = (V_{TR}, E_{TR}, w_{TR})$ に対し，提案解法 **DD** の Phase 2 を適用する.

4.6.2 関数 Div_Reg()

関数 **Div_Reg()** は，指定点集合 $P \subseteq V_{TR}$，track graph $TR = (V_{TR}, E_{TR})$ が与えられたとき，$V_{TR} = \bigcup_{1 \le i \le 4} V^i_{TR}$ なる track graph の族 $\mathcal{TR} = \{TR^i = (V^i_{TR}, E^i_{TR}, w^i_{TR}) \mid 1 \le i \le 4\}$ を求めるものである．関数 **Div_Reg()** の説明のために，いくつかの定義を与える．

$1 \le cl_x, cl_y \le n$ とする．$\langle cl_x, cl_y \rangle$ における track graph TR の族 $\mathcal{TR}(cl_x, cl_y)$ を $\mathcal{TR}(cl_x, cl_y) = \{TR^i = TR[V^i] \mid 1 \le i \le 4\}$ とおく．但し，

$V_a = \{v \in V_{TR} \mid mix_x(V_{TR}) \le x_v \le cl_x\}$,
$V_b = \{v \in V_{TR} \mid cl_x + 1 \le x_v \le max_x(V_{TR})\}$,
$V_c = \{v \in V_{TR} \mid mix_y(V_{TR}) \le y_v \le cl_y\}$,
$V_d = \{v \in V_{TR} \mid cl_y + 1 \le x_v \le max_x(V_{TR})\}$,
$V^1 = V_a \cap V_c$, $V^2 = V_c \cap V_b$, $V^3 = V_a \cap V_d$,
$V^4 = V_b \cap V_d$

である．また，$\langle cl_x, cl_y \rangle$ における TR の $Deviation(TR)$ を次のように定義する：

$Deviation(TR) = \sum_{i=1}^{4} |Ave(TR) - Dens(TR^i)|$,
$Dens(TR^i) = \frac{|P^i|}{|V^i_{TR}|}$, $Ave(TR) = \frac{1}{4} \sum_{i=1}^{4} Dens(TR^i)$.

この関数は以下の (1)，(2) を定数回繰り返す:(1) まず，$1 \le cl_x, cl_y \le n$ において上述の V_a, V_b, V_c, V_d により定義される，集合対 V_a, V_b と，集合対 V_c，V_d に対し，頂点集合 V_j に占める指定点集合の割合 $(\frac{|P_j|}{|V_j|})$ の差が大きい集合対 V_j, V_k $(\{j,k\} = \{\{a,b\},\{c,d\}\})$ を選ぶ．(2)$Deviation(TR)$ が小さくなるような cl_x，cl_y を選び，V'_j，V'_k を求める．

以下，この関数の詳細を記す.

関数 **Div_Reg**(P, TR)

step 1 $cl_x \leftarrow mid_x(TR)$, $cl_y \leftarrow mid_y(TR)$.

step 2 各 $j \in \{a,b,c,d\}$ に対し，$P_j = P \cap V_j$, $CD_j = \frac{|P_j|}{|V_j|}$ とする．

step 3 以下を定数回行う．

- $|CD_a - CD_b| < |CD_c - CD_d|$ ならば，以下を行う．/* cl_x を移動 */
  (1) $\langle cl_x, cl_y \rangle, \langle cl_x + 1, cl_y \rangle, \langle cl_x - 1, cl_y \rangle$ それぞれの場合における TR の $Deviation(TR)$ のうちで最小となる $\langle cl'_x, cl'_y \rangle$ を選ぶ．
  (2) $\mathcal{TR} \leftarrow \mathcal{TR}(cl'_x, cl'_y)$ とする．
- $|CD_a - CD_b| \ge |CD_c - CD_d|$ ならば，以下を行う．/* cl_y を移動 */
  (1) $\langle cl_x, cl_y \rangle, \langle cl_x, cl_y + 1 \rangle, \langle cl_x, cl_y - 1 \rangle$ それぞれの場合における TR の $Deviation(TR)$ のうちで最小となる $\langle cl'_x, cl'_y \rangle$ を選ぶ．
  (2) $\mathcal{TR} \leftarrow \mathcal{TR}(cl'_x, cl'_y)$ とする．

5 実験概要と結果

5.1 実験概要

既存近似解法と提案解法を計算機 (CPU:PentiumIII 800MHz,OS:FreeBSD4.3) 上の C 言語で実装し，比較実験を行った．

[**入力データ**] 入力データは $n \times n$ 矩形格子グラフを基本とし，$50 \le n \le 2000$，指定点数 p は $50 \le p \le 15000$ とした．指定点の座標は乱数によって与えた．また，禁止領域の数 m は $20 \le m \le 1500$ を満し，その大きさは 20 または 100 以下の整数 M, M' を乱数によって定め，$M \times M'$ とした．また，乱数により基準点の座標 $\langle o_x, o_y \rangle$ を与え，$o_x \le x \le o_x + M$, $o_y \le y \le o_y + M'$ の領域を禁止領域とした．

[**評価基準**] 各手法の解を評価する基準として，$ratio = \frac{|T_1| - |T_{APP}|}{|T_1|} \times 100$ （$|T_1|$: 最小スパニング木 T_1 の重み，$|T_{APP}|$: 近似解の重み）とする．さら

表 1: $\mathcal{D}=\emptyset$ なる場合の改良手法の実験結果

$\lvert P\rvert$ / n	200 / 1000		500 / 1000		800 / 1000	
	ratio	time	ratio	time	ratio	time
BR	10.5	21.41	10.5	649.0	10.5	4086
FK	9.31	0.094	9.32	0.63	9.31	1.608
FBRK	10.9	21.62	10.9	650.3	10.9	4089
BRn	10.7	50.36	10.7	1645	10.8	10278
FKn	9.42	0.233	9.45	1.16	9.43	3.110

表 2: $\mathcal{D}\neq\emptyset$ なる場合の H と $TR(H)$ の比較

	$\mathbf{BR}_r$		$\mathbf{BR}_p$		$\mathbf{BR}_{r+p}$		**DR**	
	ratio	time	ratio	time	ratio	time	ratio	time
H	10.1	97	10.1	199	10.0	359	3.99	0.75
$TR(H)$	10.2	78	10.1	162	10.1	304	4.02	0.64

表 3: $\mathcal{D}\neq\emptyset$ なる場合に対する実験結果 1

n / p	50(18%) / 50		100(13%) / 100	
	ratio	time	ratio	time
DR	4.01	0.83	4.73	3.42
$\mathbf{BR}_r$	7.80	88.4	10.4	1504
$\mathbf{BR}_p$	8.87	156	11.3	5177
$\mathbf{BR}_{r+p}$	8.90	338	12.4	6283
$\mathbf{FK}_r$	7.62	102	9.23	1460
$\mathbf{FK}_p$	7.56	164	9.12	2276
$\mathbf{FK}_{r+p}$	7.62	368	9.08	3765
BRG	7.95	3300	11.0	113441
KM	2.66	0.28	2.71	1.7
DD	3.69	0.45	3.96	2.8

表 4: $\mathcal{D}\neq\emptyset$ なる場合に対する実験結果 3

n / p	50(31%) / 200		100(30%) / 500	
	ratio	time	ratio	time
DR	0.86	1.57	9.30	10.8
$\mathbf{BR}_r$	2.63	254	17.3	7289
$\mathbf{BR}_p$	2.65	536	16.7	15752
$\mathbf{BR}_{r+p}$	2.65	1097	17.7	31440
$\mathbf{FK}_r$	1.59	450	13.8	9032
$\mathbf{FK}_p$	1.59	796	14.9	16461
$\mathbf{FK}_{r+p}$	1.59	1715	14.2	26379
BRG	2.61	2696	–	–
KM	0.39	0.34	4.46	3.9
DD	0.63	1.07	9.04	9.59

表 5: $\mathcal{D}\neq\emptyset$ なる場合に対する実験結果 2

n / p	300(7%) / 300		500(5%) / 500		1000(3%) / 1000		2000(3%) / 2000	
	ratio	time	ratio	time	ratio	time	ratio	time
DR	4.59	370	4.71	2383	4.43	29176	4.56	128833
KM	3.01	325	3.13	1880	2.97	26958	3.01	110931
DD	3.98	339	4.35	1964	4.07	27832	4.12	120045

表 6: $\mathcal{D}\neq\emptyset$ なる場合に対する実験結果 4

n / p	300(31%) / 5000		500(21%) / 10000		1000(10%) / 10000		1000(13%) / 15000	
	ratio	time	ratio	time	ratio	time	ratio	time
DR	10.4	1347	7.50	9389	6.21	82131	7.27	82801
KM	5.23	1072	3.55	5352	3.97	49356	4.24	66374
DD	8.65	1209	6.21	6063	6.01	56215	6.99	71002

に，入力データの禁止領域密集度の基準として，全頂点数，禁止領域に含まれる頂点数をそれぞれ $|N|$, $|D|$ としたとき，$\mathrm{OB\_dens}=\dfrac{|T_1|}{|N|-|D|}\times 100$ を用いる．また，結果表の – は 1 つのデータに対して，50 時間かけても解が求まらなかったことを示す．

5.2 結果と考察

$\mathbf{BR}_r$，$\mathbf{BR}_p$，$\mathbf{BR}_{r+p}$，**DR**（track graph を用いる）の結果と，それぞれを track graph のかわりに H を用いた場合の結果との ratio，計算時間 time（単位：秒）を比較した実験結果を図 2 に示す．この実験結果より，track graph を用いることで計算時間を短縮でき，求まる解の精度も同等かそれ以上であること，すなわち track graph の有用性がわかる．

BR を拡張した手法 $\mathbf{BR}_r$，$\mathbf{BR}_p$，$\mathbf{BR}_{r+p}$ は **BRG** と同等の解精度を持ち，**BRG** よりも計算時間が早い．**BRG** では，$\mathbf{BR}_{r+p}$ よりも 3 点に対する最小の格子スタイナー木を多く求める可能性があるが *ratio* は拡張手法 $\mathbf{BR}_{r+p}$ よりも悪くなる場合がある．このことから，3 点に対する最小の格子スタイナー木の置き換えが，必ずしも解精度の向上につながるとは限らないことがわかる．

拡張手法 $\mathbf{BR}_r$，$\mathbf{BR}_p$，$\mathbf{BR}_{r+p}$ と提案手法 **DD** を比較すると ratio では $\mathbf{BR}_{r+p}>\mathbf{BR}_p=\mathbf{BR}_r>\mathbf{DD}$，計算時間は $\mathbf{BR}_{r+p}>\mathbf{BR}_p>\mathbf{BR}_r\gg\mathbf{DD}$ であった．したがって，計算時間の非常に短い **DD** を領域分割手法に組み込むことは，高速化に有効であると考えられる．また，$\mathbf{BR}_r$ を分割手法に組み込むことで，計算時間は長くなるが精度の改善が期待できる．

DR は，**BRG** や拡張手法 $\mathbf{BR}_{r+p}$ に比べると ratio は悪いが計算時間が非常に短い．**BRG** や拡張手法 $\mathbf{BR}_r$，$\mathbf{BR}_p$，$\mathbf{BR}_{r+p}$ は格子領域が 300×300 になると計算時間が非常に大きくなり，実用的ではないことから考えると，**DR** は有用と思われる．

参考文献

[1] P. Berman and V. Ramaiyer, Improved approximation algrithms for the Steiner tree problem. J. Algoritms, Vol. 17, pp. 381–408, 1994.

[2] M. R. Garey and D. S. Johnson, The rectilinear Steiner problem is NP–complete. SIAM J. Appl. Math, Vol. 32, pp. 826–834, 1977.

[3] F. K. Hwang, An $O(n\log n)$ algorithm for suboptimal rectilinear Steiner trees. IEEE Trans. Circuits and Systems, Vol. CAS-26, pp. 75–77, 1979.

[4] F. K. Hwang, On Steiner minimal trees with rectilinear distance. SIAM J. Appl. Math, Vol. 30, pp. 104–114, 1976.

[5] Y. F. Wu, P. Widmayer, M. D. F. Schlag, C. K. Wong, Rectilinear shortest paths and minimum spanning trees in the presence of rectilinear obstacles. IEEE Trans. Comput. , Vol. C-36(3), pp. 321–331, 1987.

[6] K.Mehlhorn, A faster approximation algorithm for the Steiner problem in graphs, Inf. Process. Lett. Vol. 27, pp. 125–128, 1988.

[7] M. Goda, N. Toyama and T. Watanabe, A Parallel Detailed Router TRED, 6th TRANSPUTER/OCCAM JAPAN, IOS Press, Amsterdam, pp. 156–169, 1994.