配送時間制約を考慮したネットワーク設計問題

Network design problems with delivery time constraints

片山 直登：流通経済大学 流通情報学部 教授

略　歴

1982年 早稲田大学理工学部卒業、1988年　同大学院博士後期課程単位取得退学。博士（物流情報学）。早稲田大学理工学部助手、金沢工業大学工学部講師、1996年 流通経済大学流通情報学部助教授を経て、2002年から現職。

［要約］　インターネット通販の進展に伴い、宅配便のような企業から個人への配送量が急増し、適切な配送サービスを維持するためには、施設や輸送能力の拡充など根本的なネットワークの再構築が必要となってきている。本研究では、当日配送、翌日配送などの複数の配送時間制約と配送車である資源のバランスを考慮する条件を考慮した資源の配置とフローを求めるアーク設計問題のモデルを扱う。このモデルに対して、配送時間制限の考慮方法の異なるいくつかの定式化、およびフローや資源の表現が異なるいくつかの定式化を提示し、ベンチマーク問題を用いてモデルと定式化による求解の困難性を明らかにする。

キーワード　ネットワーク設計問題、配送時間制約、資源バランス

1　はじめに

インターネット通販の進展に伴い、宅配便のような企業から個人への配送量が急増している。人手不足や再配送問題も加わり、ラストワンマイルとよばれる配送センターと顧客間の配送は大きな問題となっている。また、当日配送、翌日配送などの短納期配送サービスが一般化している。加えて、センター間の幹線輸送では輸送能力がひっ迫する状況にある。今後、適切な配送サービスを維持するためには、施設や輸送能力の拡充など根本的なネットワークの再構築が必要となってきている。しかし、ITの進展により膨大な量のビッグデータを把握できる状態であるにも関わらず、企業は日々の大量の荷物の処理業務の対応のため、根本的な配送・輸送ネットワークの見直しが進まない状況となっている。

配送・輸送などのネットワークの構成は大きく3つのレベルに分類することができる。配送センターなどの施設の適切な配置と能力設定を決める問題、施設間の輸送路と荷物の輸送経路を決める問題、および企業・施設間や施設・顧客間の配送経路を決める問題である。配置と能力設定を決める問題は施設であるノードの配置とフローを求めるノード設計問題、施設間の輸送路と荷物の輸送経路を決める問題はノード間の資源の配置とフローを求めるアーク設計問題、企業･施設間や施設･

顧客間の配送経路を求める問題は配送・集荷経路問題として表すことができる。これらを同時に取り扱うことが望ましいが、問題の規模や複雑性や対象となる期間が異なるなどの点から、同時に取り扱うことは困難が伴う。

一般的に、ネットワーク設計問題はアークの設計問題として扱われることが多い(Crainic et al. 2021)。このアーク設計のモデルでは、ノードをノード間のアークに変換することにより、ノードの設計問題を扱うことが可能である。ネットワーク設計問題には、アークの容量を考慮する／しないモデル(Crainic et al. 2021)、資源のバランスやスケジュールを考慮するモデル（Jarrah et al. 2009, Andersen et al. 2011, Erera et al. 2013, Crainic et al. 2016, Boland et al. 2017)、フローが1つのパスを通るモデル（Atamtürk and Rajan 2002, Yaghini and Kazemzadeh 2012, Li et al. 2017）や確率的モデル(Crainic et al. 2011, Bai et al. 2014、Rahmaniani et al. 2018、Wang et al. 2019）など様々なモデルが研究されている。

資源のバランスやスケジュールを考慮するモデルは配送車などの資源の巡回経路を考慮するモデルであり、時間空間ネットワークにより資源の巡回スケジュールを求めることができる。一方、ネットワークの構成が与えられたもとで、配送期限を条件として荷物の流れであるフローを求めるモデルは、コンテナの海上輸送である定期船輸送の分野で数多くの研究（Karsten et al. 2017, Balakrishnan and Karsten 2017, Tierney et al. 2019, Koza et al. 2020）が行われている。また、Trivella et al.（2021）は配送期限を超える需要に対してペナルティ費用を付加するソフト配送時間モデルを提案している。

一方、配送時間の制限を取り扱ったネットワーク設計モデルはそれほど多くはなく、Hellsten et al.（2021）は単一の配送期限の制限を考慮したモデルに対して、いくつかの定式化を示し、列生成法と分枝価格法を用いた解法を示している。

本研究では、資源の配置とフローを求めるアーク設計問題を対象とし、当日配送、翌日配送などの複数の配送時間制約を考慮したネットワーク設計問題を扱う。加えて、配送車である資源のバランスを考慮する資源バランス設計条件と同一品種のフローが1つのパスを流れる条件を考慮する。このようなモデルに対して、配送時間制限の考慮方法の異なるいくつかの定式化、およびフローや資源の表現が異なるいくつかの定式化を提示する。ベンチマーク問題を用いて、いくつかの定式化について数値実験を行い、問題の条件や定式化による求解の困難性を明らかにする。

2　モデルの前提条件

はじめに、本研究における前提条件を列挙する。

- ノード集合が与えられる。
- 向きをもつアーク候補集合が与えられる。
- アーク上には整数個の資源が割り当てられる。
- 資源には決められた容量が与えられる。

- 品種集合が与えられる。
- 品種ごとの需要量が与えられ、品種は異なる始点・終点をもち、始点・終点の対で表す。
- アーク上を流れるフローに対して、単位当たりのフロー費用が与えられる。
- アーク上を流れる品種のフローの移動時間が与えられる。
- 同一の終点をもつ品種の需要は、単一の経路上を移動する。

問題により、以下の前提条件のうちのいくつかを満足する。

- 資源はアークで構成されるサイクル上に配置される。
- アーク上を使用する資源に対して、単位当たりの固定費用であるデザイン費用が与えられる。
- 使用する資源に対して、固定費用が与えられる。

配送時間に関しては、次の前提条件のうちいずれかを満足する。

- 複数の配送時間の制限が与えられ、それぞれの配送時間の制限を満たす需要のカバー率の下限が与えられる。
- 複数の配送時間の制限が与えられ、それぞれの配送時間の制限を超える需要については、超過時間に比例したペナルティ費用が発生する。

配送時間の制限には、当日、翌日など複数の時間制限があるものとする。ここでは、配送時間の制限について2つのモデルを使用する。一つ目は、複数の時間制限を満たす需要のカバー率を与えるモデルである。例えば、全需要の10%までを当日に配送し、全需要の50%までを翌日までに配送するというような条件を考慮するモデルである。二つ目は、時間制限を超える需要に対してはペナルティ費用が発生するモデルである。ここでは、非減少の区分的線形関数をペナルティ費用の関数とする。

モデルの定義は以下の通りである。複数種類のモデルを扱うため､共通部分を記述する。

ノード集合、アーク候補集合、アークに割り当てる容量・費用をもつ資源集合、始点と終点をもつ品種の需要、各品種の配送時間制限集合が与えられる。フローの単一パス条件を満たす。このとき、全体の費用合計を最小にするフローおよびアークに割り当てる資源を求めよ。

3　配送時間制限カバー率モデル

はじめに、配送時間の制限を全需要に対するカバー率を用いる4つのモデルを示す。4つのモデルは、整数デザイン変数とアークフロー変数を用いたモデル、0-1デザイン変数とアークフロー変数を用いたモデル、整数デザイン変数とパスフロー変数を用いたモデル、整数デザイン変数とアーク通過時間フロー変数を用いたモデルである。なお、デザイン変数はアークに配置される資源数を表す変数の総称である。

3.1　整数デザイン・アークフローモデル

デザイン変数値が0以上の整数値を取り、アークフロー変数を用いたネットワーク設計

モデルを示す。

ノード集合をN、向きのあるアーク候補集合をA、品種集合をKとする。ノードnを始点とするアークの終点であるノード集合をN_n^+、ノードnを終点とするアークの始点であるノード集合をN_n^-とする。配送時間制限の集合をHとする。また、0以上の整数の集合をZとする。

品種kは始点o^k、終点d^kをもち、需要量q^kが与えられ、品種の需要はいくつかのアークを経由して始点o^kから終点d^kに流れるものとする。最大の配送時間制限をh_{max}とし、配送時間制限集合Hは$H = \{1, \cdots, h_{max}\}$で表される。$h$番目の配送時間制限を$l_h$としたとき、$l_{h-1} < l_h (h = 2, \cdots, h_{max})$を満たす。アーク$(i,j)$には、アーク上の単位当たりのフローに対するフロー費用c_{ij}、アークに割り当てられる資源に対する単位当たりのデザイン費用f_{ij}、アーク上の移動時間t_{ij}、アーク上の資源の単位当たりの容量b_{ij}が与えられる。ノードnが品種kの始点o^kであれば−1、終点d^kであれば1、それ以外であれば0である定数をe_n^kとする。また、h番目の配送時間制限を満たす需要の全需要に対する需要カバー率をd_hとする。ただし、$d_{h-1} < d_h$（$h=2, \cdots, h_{max}$）とする。

アーク(i,j)に対して、品種kのフローがアーク上を流れるとき1、そうでないとき0であるフロー変数をx_{ij}^k、アーク上に割り当てられる資源数を表す整数変数である資源数変数をy_{ij}とする。また、品種kの需要がh番目の配送時間制限を満たすとき1、そうでないとき0である時間制限変数をv_h^kとする。

このとき、整数デザイン・アークフローモデル*IDAF*は次のようになる。

$IDAF$：

$$minimize \quad \sum_{(i,j)\in A}\sum_{k\in K} q^k c_{ij} x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{1}$$

$subject\ to$

$$\sum_{i\in N_n^+} x_{in}^k - \sum_{j\in N_n^-} x_{nj}^k = e_n^k \quad \forall n \in N, k \in K, \tag{2}$$

$$\sum_{k\in K} q^k x_{ij}^k \le b_{ij} y_{ij} \quad \forall (i,j) \in A, \tag{3}$$

$$x_{ij}^k \le y_{ij} \quad \forall k \in K, (i,j) \in A, \tag{4}$$

$$\sum_{(i,j)\in A} t_{ij} x_{ij}^k \le l_{max} - (l_{max} - l_h) v_h^k \quad \forall h \in H, k \in K, \tag{5}$$

$$\sum_{k\in K} q^k v_h^k \ge d_h \sum_{k\in K} q^k \quad \forall h \in H, \tag{6}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k \in K, (i,j) \in A, \tag{7}$$

$$v_h^k \in \{0,1\} \quad \forall h \in H, k \in K, \tag{8}$$

$$y_{ij} \in Z \quad \forall (i,j) \in A. \tag{9}$$

（1）式は総費用を表す目的関数であり、これを最小化する。第一項はアーク上のフロー費用の合計であり、第二項はアーク上のデザイン費用の合計である。y_{ij}は整数値をとる資源数変数であり、アーク上のデザイン費用は資源数変数に比例して発生する。（2）式はフ

ローの保存式であり、始点を発した需要のフローがいくつかのアークを経由して終点に達することを表す。xが0-1変数であることから、特定の品種が流れるパスは1 本に限定される。一般のネットワーク設計モデルでは複数のパスを許す場合が多いが、複数パスを許すモデルに拡張することは容易である（Hellsten et al. 2021）。（3）式は容量制約式であり、左辺はアーク上を流れるフローの合計、右辺は資源数と容量の積であるアーク上の容量である。（4）式は強制制約式であり、アーク上に資源が設定されるときに限り、アーク上をフローが流れることを表す。一般的には強い制約式となるが、この定式化では資源数変数が整数値をとるため、一般的なモデルと比べると弱い式となる。（5）式は配送時間制限に関する式である。左辺は品種kの始点・終点間の配送時間を表す。右辺は、v_h^kが1のときにl_hとなりh番目の制限時間を満足し、v_h^kが0のときにl_{max}となり最大の制限時間を満足することを表す。（6）式は配送時間制限を満足する需要が全需要に対するカバー率を表す式である。左辺は配送時間がh番目の制限時間を満たす需要の合計であり、右辺は満たさなければならない量である。（7）式と（8）式は変数の0-1条件であり、（9）式は変数の整数条件である。

3.2　0-1デザイン・アークフローモデル

デザイン変数値が0-1値を取り、アークフロー変数を用いたネットワーク設計モデルを示す。

アーク上の資源を表す整数の資源数変数を0-1変数に分解することにより、より強い定式化を行うことができる。アーク（i,j）に配置できる資源の集合をS_{ij}とする。アーク（i,j）上にs個の資源が配置されるとき1、そうでないとき0を表す0-1のデザイン変数をy_{ij}^sとおく。また、アーク（i,j）上のs個目の資源を使用する品種kのフロー比率を表す連続変数であるアーク資源数フロー変数を$u_{ij}^{k,s}$とおく。

このとき、0-1デザイン・アークフローモデル*BDAF*は次のようになる。

$BDAF$：

$$minimize \quad \sum_{(i,j)\in A}\sum_{k\in K} q^k c_{ij} x_{ij}^k + \sum_{(i,j)\in A}\sum_{s\in S_{ij}} s f_{ij} y_{ij}^s \tag{10}$$

$subject\ to$

$$\sum_{i\in N_n^+} x_{in}^k - \sum_{j\in N_n^-} x_{nj}^k = e_n^k \quad \forall n\in N, k\in K, \tag{11}$$

$$x_{ij}^k = \sum_{s\in S_{ij}} u_{ij}^{ks} \quad \forall k\in K, (i,j)\in A, \tag{12}$$

$$\sum_{k\in K} q^k u_{ij}^{ks} \geq (s-1)\, b_{ij} y_{ij}^s \quad \forall s\in S_{ij}, (i,j)\in A, \tag{13}$$

$$\sum_{k\in DK} q^k u_{ij}^{ks} \leq s\, b_{ij} y_{ij}^s \quad \forall s\in S_{ij}, (i,j)\in A, \tag{14}$$

$$\sum_{(i,j)\in A} t_{ij} x_{ij}^k \leq l_{max} - (l_{max} - l_h) v_h^k \quad \forall h\in H, k\in K, \tag{15}$$

$$\sum_{k\in K} q^k v_h^k \geq d_h \sum_{k\in K} q^k \quad \forall h \in H, \tag{16}$$

$$u_{ij}^{ks} \leq y_{ij}^s \quad \forall k \in K, s \in S_{ij}, (i,j) \in A, \tag{17}$$

$$\sum_{s\in S_{ij}} y_{ij}^s \leq 1 \quad \forall (i,j) \in A, \tag{18}$$

$$u_{ij}^{ks} \geq 0 \quad \forall k \in K, s \in S_{ij}, (i,j) \in A, \tag{19}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k \in K, (i,j) \in A, \tag{20}$$

$$v_h^k \in \{0,1\} \quad \forall h \in H, k \in K, \tag{21}$$

$$y_{ij}^s \in \{0,1\} \quad \forall s \in S_{ij}, (i,j) \in A. \tag{22}$$

(10) 式は総費用を表す目的関数であり、これを最小化する。デザイン費用は資源数に比例して発生し、第二項はアーク上に配置されるs個の資源のデザイン費用の合計となる。(12)式はアークフローとアーク資源数フローの関係式である。左辺はこのアークを使用する品種のフローの比率であり、右辺はこのアークを使用する品種の資源に対するフロー比率の合計を表す。(13) 式は容量の下限の制約式であり、s個の資源を使用するアーク上のフローの合計は$s-1$個の資源数を配置したときの容量以上であることを表す。(14) 式は容量の上限の制約式であり、s個の資源を使用するアーク上のフローの合計はs個の資源数を配置したときの容量以下であることを表す。(17) 式は強制制約式であり、アーク上にs個目の資源が設定されるときに限り、アーク上を資源sのアーク資源数フローが流れることを表す。左辺は0または1をとり、右辺も0または1をとることになり、強い制約式となる。(19) 式は変数の非負条件である。

この定式化は区分的線形費用をもつネットワーク設計問題で用いられている(Croxton et al. 2003)。この定式化は、アーク容量の強い上限値が与えられる場合に有効なものとなる。一方、アークに配置できる資源数に上限がない場合は、使用する0-1変数は膨大なものとなる。

3.3 整数デザイン・パスフローモデル

デザイン変数値が整数値を取り、パスフロー変数を用いたネットワーク設計モデルを示す。

品種kの取りうるパスの集合をP^kとする。品種kのパスpを使用するとき1、そうでないとき0であるパスフロー変数をz_p^kとし、パスpがアーク（i,j）を使用するとき1そうでないとき0である定数をδ_{ij}^pとする。

このとき、整数デザイン・パスフローモデル*IDPF*は次のようになる。

$IDPF$：

$$minimize \quad \sum_{(i,j)\in A} \sum_{k\in K} q^k c_{ij} \sum_{p\in P^k} \delta_{ij}^p z_p^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{23}$$

$subject\ to$

$$\sum_{p\in P^k} z_p^k = 1 \quad \forall k \in K \tag{24}$$

$$\sum_{k\in K}\sum_{p\in P^k} q^k \delta_{ij}^p z_p^k \le b_{ij} y_{ij} \quad \forall (i,j)\in A, \tag{25}$$

$$\sum_{p\in P^k} \delta_{ij}^p z_p^k \le y_{ij} \quad \forall k\in K, (i,j)\in A, \tag{26}$$

$$\sum_{(i,j)\in A} t_{ij} x_{ij}^k \le l_{max} - (l_{max} - l_h) v_h^k \quad \forall h\in H, k\in K, \tag{27}$$

$$\sum_{k\in K} q^k v_h^k \ge d_h \sum_{k\in K} q^k \quad \forall h\in H, \tag{28}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k\in K, (i,j)\in A, \tag{29}$$

$$z_p^k \in \{0,1\} \quad \forall p\in P^k, k\in K, \tag{30}$$

$$v_h^k \in \{0,1\} \quad \forall h\in H, k\in K, \tag{31}$$

$$y_{ij} \in Z \quad \forall (i,j)\in A. \tag{32}$$

（23）式は総費用を表す目的関数であり、これを最小化する。$\Sigma_{p\in P^k} \delta_{ij}^p z_p^k = x_{ij}^k$であることから、第一項はアーク上のフロー費用の合計となる。（24）式はフローの保存式であり、パスフロー変数が0-1変数であることから、品種kの取りうるパス集合P^kから1本のパスを選択することを表す。（25）式は容量制約式であり、左辺はアーク上を流れるフローの合計、右辺は資源数と容量の積であるアーク上の容量の合計である。（26）式は強制制約式であり、アーク上に資源が設定されるときに限り、アーク上をフローが流れることを表す。（30）式は変数の0-1条件である。

多くのネットワーク設計問題では、このパスフローによる定式化が使用される。取りうるパスは指数個あるために陽的に列挙することは困難である。このため、適時、目的関数値を減少させるような変数である列を生成する列生成法を使用する。生成する列を求める問題は最短経路問題または制約付きの最短経路問題となり、効率的に変数である列を生成することができる。列生成法は、線形緩和問題を最適に解くための手法であり、最適解または近似解を求めるためには、別の手法を組み合わせる必要がある。

なお、0-1デザイン変数を用いたパスフローによる定式化も可能であるが、ここでは省略する。

3.4　整数デザイン・アーク通過時刻フローモデル

デザイン変数値が整数値を取り、アーク通過時間フロー変数を用いたネットワーク設計モデル（Hellsten et al. 2021）を示す。

時刻の集合をDとし、品種kがアーク（i,j）のノードiから出る可能性のある時刻の集合をD_{ij}^kとする。また、時刻dにノードiを出るアーク（i,j）を使用する品種kのフローが存在するとき1、そうでないとき0であるアーク通過時刻フロー変数をx_{ij}^{kd}とする。

このとき、整数デザイン・アーク通過時刻フローモデル*IDATF*は次のようになる。

$IDATF$：

$$minimize \quad \sum_{(i,j)\in A}\sum_{k\in K}\sum_{d\in D_{ij}^k} q^k c_{ij} x_{ij}^{kd} + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{33}$$

$$subject\ to$$

$$\sum_{i\in N_n^+ : d-t_{in}\in D_{in}^k} x_{in}^{k,d-t_{in}} - \sum_{j\in N_n^- : d\in D_{nj}^k} x_{nj}^{kd} = 0 \quad \forall n \in N\backslash\{o^k, d^k\}, k \in K, d \in D, \tag{34}$$

$$\sum_{j\in N_{o^k}^-} x_{o^k j}^{k0} = 1 \quad \forall k \in K, \tag{35}$$

$$\sum_{i\in N_{d^k}^+}\sum_{d\in D_{id^k}^k} x_{id^k}^{kd} = 1 \quad \forall k \in K, \tag{36}$$

$$\sum_{k\in K}\sum_{d\in D_{ij}^k} q^k x_{ij}^{kd} \le b_{ij} y_{ij} \quad \forall (i,j) \in A, \tag{37}$$

$$x_{ij}^{kd} \le y_{ij} \quad \forall d \in D_{ij}^k, k \in K, (i,j) \in A, \tag{38}$$

$$\sum_{(i,j)\in A}\sum_{d\in D_{ij}^k} t_{ij} x_{ij}^{kd} \le l_{max} - (l_{max} - l_h) v_h^k \quad \forall h \in H, k \in K, \tag{39}$$

$$\sum_{k\in K} q^k v_h^k \ge d_h \sum_{k\in K} q^k \quad \forall h \in H, \tag{40}$$

$$x_{ij}^{kd} \in \{0,1\} \quad \forall d \in D_{ij}^k, k \in K, (i,j) \in A, \tag{41}$$

$$v_h^k \in \{0,1\} \quad \forall h \in H, k \in K, \tag{42}$$

$$y_{ij} \in Z \quad \forall (i,j) \in A. \tag{43}$$

（33）式は総費用を表す目的関数であり、これを最小化する。第一項は取りうるすべての時刻のフロー費用の合計である。（34）式から（36）式はフロー保存式である。（34）式は、中継ノードにおける関係式であり、取りうるすべての時刻に対して、ノードに入るフローとノードから出るフローが一致することを表す。ここで、ノードnに入るフローと出るフローにはt_{in}の時間差があることに注意する。（35）式は、時刻0に始点o^kから出るフローの合計が1であることを表す。（36）式は、そのノードで取りうる時刻について終点d^kに入るアークフローの合計が1であることを表す。

取りうる時刻は、アーク上の移動時間の公倍数ではあるが膨大なものとなるため、移動時間の丸め込みなどにより、時刻集合のサイズを縮小することが必要になる。

4　資源バランス設計モデル

これまで示したモデルは、アークに配置する資源数または容量を決めるモデルであり、資源が輸送・配送車の場合には行きの便のみを考慮し、帰り便は考慮しないモデルとなる。輸送・配送車は巡回するものとし、巡回路であるサイクルを考慮する、またはノードを出入りする資源数のバランスを考慮するモデルは資源バランス設計モデル（Pedersen et al. 2009）またはサービスネットワーク設計問題（Andersen et al. 2009）とよばれている。ネットワークを時間・空間ネットワークとすることにより、資源の巡回スケジュールを求めることも可能となる。

4.1　資源バランス・0-1 デザイン・パスフローモデル

デザイン変数値が整数値を取り、パスフローを用いた資源バランス設計モデルを示す。

資源の集合をR、資源rが使用されるときの固定費用をg^rとする。資源rが使用されるとき1、そうでないとき0である資源変数をw^rとする。資源rの最大稼働時間をTとする。また、アーク（i,j）上で、資源rが使用されるとき1、そうでないとき0である資源デザイン変数をy_{ij}^rとする。資源rのタイプが異なり、容量や移動時間が異なってもよいが、ここではすべての資源は同じタイプとする。

このとき、整数パスフロー変数を用いた資源バランス設計モデル$ABDAF$は次のようになる。

$ABDAF$：

$$minimize \quad \sum_{(i,j)\in A}\sum_{k\in K}\sum_{p\in P^k} q^k c_{ij}\delta_{ij}^p z_p^k + \sum_{(i,j)\in A}\sum_{r\in R} f_{ij}y_{ij}^r + \sum_{r\in R} g^r w^r \tag{44}$$

$$subject\ to$$

$$\sum_{p\in P^k} z_p^k = 1 \quad \forall k \in K \tag{45}$$

$$\sum_{k\in K}\sum_{p\in P^k} q^k \delta_{ij}^p z_p^k \le b_{ij}\sum_{r\in R} y_{ij}^r \quad \forall (i,j)\in A, \tag{46}$$

$$\sum_{p\in P^k} \delta_{ij}^p z_p^k \le \sum_{r\in R} y_{ij}^r \quad \forall k\in K, (i,j)\in A, \tag{47}$$

$$\sum_{(i,j)\in A} t_{ij}x_{ij}^k \le l_{max} - (l_{max}-l_h)v_h^k \quad \forall h\in H, k\in K, \tag{48}$$

$$\sum_{k\in K} q^k v_h^k \ge d_h \sum_{k\in K} q^k \quad \forall h\in H, \tag{49}$$

$$\sum_{i\in N_n^+} y_{in}^r - \sum_{j\in N_n^-} y_{nj}^r = 0 \quad \forall n\in N, r\in R \tag{50}$$

$$\sum_{(i,j)\in A} y_{ij}^r \le |A| w^r \quad \forall r\in R, \tag{51}$$

$$\sum_{(i,j)\in A} t_{ij}y_{ij}^r \le T w^r \quad \forall r\in R, \tag{52}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k\in K, (i,j)\in A, \tag{53}$$

$$v_h^k \in \{0,1\} \quad \forall h\in H, k\in K, \tag{54}$$

$$y_{ij}^r \in \{0,1\} \quad \forall r\in R, (i,j)\in A. \tag{55}$$

（44）式は総費用を表す目的関数であり、これを最小化する。第二項はアーク上で使用する資源デザイン費用の合計である。第三項はネットワーク上で使用する資源に対して発生する固定費用の合計である。（46）式は容量制約式であり、右辺はアークで使用される資源の合計数と容量の積であるアーク上の容量である。（47）式は強制制約式であり、アー

ク上にいずれかの資源が設定されるときに限り、アーク上をフローが流れることを表す。(50)式は資源バランス式であり、資源rがノードに入ればそのノードから出ることになり、資源rがノードに入らなければそのノードから出ないことを表す。バランス式により、資源はネットワーク上の巡回路で使用されることになる。(51)式は資源が使われるか否かを判定する式であり、いずれかの資源デザイン変数が1であれば、その資源が使用されることを表す。なお、左辺の最大数はアーク数となるため、右辺にはアーク数を係数として使用している。(52)式は資源の稼働時間が最大稼働時間以下であることを表す。

4.2 資源サイクル・0-1 デザイン・アークフローモデル

資源バランス設計モデルでは、資源が巡回路であるサイクルを形成することになる。そこで、個々のアーク上のデザイン変数としてではなく、取りうるサイクルをデザイン変数としても定式化することができる(Pedersen et al. 2009)。ここでは、資源サイクル変数を用いた資源バランス設計モデルを示す。

資源サイクルはあるノードからいくつかのアークとノードを経て、このノードに戻るサイクルであり、このサイクル上を資源が巡回することを意味する。資源サイクルが選択されると、この資源サイクル上のアークに容量が設定される。

取りうる資源のサイクルの集合をMとする。なお、Mには同じサイクルが複数含まれても良いものとする。資源サイクルmを使用するとき1、そうでないとき0である資源サイクル変数をg_mとし、資源サイクルmを使用する際に発生する固定費用をw_mとする。また、資源サイクルmがアーク(i,j)を使用するとき1、そうでないとき0である定数をϵ_{ij}^{m}とする。

このとき、資源サイクル変数を用いた資源バランス設計モデル$CYBD$は次のようになる。

$CYBD$:

$$minimize \quad \sum_{(i,j)\in A}\sum_{k\in K} q^k c_{ij} x_{ij}^k + \sum_{(i,j)\in A}\sum_{m\in M} \epsilon_{ij}^m f_{ij} w_m + \sum_{m\in M} g_m w_m \tag{56}$$

$$subject\ to$$

$$\sum_{i\in N_n^+} x_{in}^k - \sum_{j\in N_n^-} x_{nj}^k = e_n^k \quad \forall n\in N, k\in K, \tag{57}$$

$$\sum_{k\in K} q^k x_{ij}^k \le b_{ij} \sum_{m\in M} \epsilon_{ij}^m w_m \quad \forall (i,j)\in A, \tag{58}$$

$$x_{ij}^k \le \sum_{m\in M} \epsilon_{ij}^m w_m \quad \forall k\in K, (i,j)\in A, \tag{59}$$

$$\sum_{(i,j)\in A} t_{ij} x_{ij}^k \le l_{max} - (l_{max} - l_h) v_h^k \quad \forall h\in H, k\in K, \tag{60}$$

$$\sum_{k\in K} q^k v_h^k \ge d_h \sum_{k\in K} q^k \quad \forall h\in H, \tag{61}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k \in K, (i,j) \in A, \tag{62}$$
$$v_h^k \in \{0,1\} \quad \forall h \in H, k \in K, \tag{63}$$
$$w_m \in \{0,1\} \quad \forall m \in M. \tag{64}$$

（56）式は総費用を表す目的関数であり、これを最小化する。第二項はアーク上で使用する資源サイクルがアークを使用するときのデザイン費用の合計であり、第三項は使用された資源サイクルに対して発生する固定費用の合計である。（58）式は容量制約式であり、右辺はアークで使用される資源サイクルの合計数と容量の積であるアーク上の容量である。（59）式は強制制約式であり、アーク上にいずれかの資源サイクルが設定されるときに限り、アーク上をフローが流れることを表す。資源サイクル変数は資源バランス条件と稼働時間制限を満足するものに限るため、資源バランス式と稼働時間制限式は必要ない。

資源サイクルは指数個あることから、資源サイクル変数は列生成法により列挙する。

5　配送超過時間ペナルティーモデル

配送時間制限を超過する需要に対して、ペナルティーを与えるモデルを示す。ここでは、整数デザイン変数とアークフロー変数を用いたモデルを示す。

h番目の時間制限l_hを超えたときの品種kに対する単位当たりの非負のペナルティ費用をp_h^kとし、時間制限l_hの超過時間をs_h^kとする。

このとき、整数デザイン変数とアークフロー変数を用いた配送超過時間ペナルティーモデルOTPは次のようになる。

（65）式は総費用を表す目的関数であり、

OTP：

$$minimize \quad \sum_{(i,j)\in A}\sum_{k\in K} q^k c_{ij} x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} + \sum_{k\in K}\sum_{h\in H} p_h^k s_h^k \tag{65}$$

$subject\ to$

$$\sum_{i\in N_n^+} x_{in}^k - \sum_{j\in N_n^-} x_{nj}^k = e_n^k \quad \forall n \in N, k \in K, \tag{66}$$
$$\sum_{k\in K} q^k x_{ij}^k \le b_{ij} y_{ij} \quad \forall (i,j) \in A, \tag{67}$$
$$x_{ij}^k \le y_{ij} \quad \forall k \in K, (i,j) \in A, \tag{68}$$
$$s_h^k \ge \sum_{(i,j)\in A} t_{ij} x_{ij}^k - l_h \quad \forall h \in H, k \in K, \tag{69}$$
$$s_h^k \ge 0 \quad \forall h \in H, k \in K, \tag{70}$$
$$x_{ij}^k \in \{0,1\} \quad \forall k \in K, (i,j) \in A, \tag{71}$$
$$y_{ij} \in Z \quad \forall (i,j) \in A. \tag{72}$$

これを最小化する。第三項は配送時間超過によるペナルティー費用の合計であり、配送時間制限に対する単位当たりのペナルティー費用と超過時間の積で表す。（69）式は、品種kの配送時間期限l_hに対する超過時間である。品種kの配送時間が制限時間を超えなければ

左辺は負となり、(70) 式からs_h^kは0となる。また、配送時間が時間期限を超えた場合、左辺は非負の超過時間となる。ペナルティ費用は非負であり、目的関数値を最小化することから、(69) 式は等号で成り立ち、s_h^kは超過時間に一致する。

パスフロー変数や0-1デザイン変数を用いた定式化や、資源バランスを考慮した定式化も可能である。

6　数値実験

配送時間制約をもつネットワーク設計問題で用いられるベンチマーク問題を用いて、モデルと定式化による問題の困難性を比較する。

使用したベンチマーク問題は、Hellsten et al. (2021) が用いたインスタンスの中のR問題である。このインスタンスは、容量制約をもつネットワーク設計問題のためにCrainic et al. (2001) が提示したものに、アーク上の移動時間を追加したものである。このR問題は、r04からr18（r06は欠）に分類され、それぞれは9個のインスタンスで構成される。表1にノード数、アーク候補数、品種数を示す。また、表2 に容量と費用のレベルを示し、c1はアーク容量が大きい、c8はアーク容量が小さいことを表し、f01はデザイン費用が相対的に安い、f10はデザイン費用が相対的に高いことを表す。

計算の対象とした定式化は、整数デザイン・アークフローモデル*IDAF*、0-1デザイン・アークフローモデル*BDAF*、および資源バランス・0-1デザイン・アークフローモデル*ABDAF*である。*ABDAF*では、資源のアークデザイン費用を考慮するが固定費用を考慮しないモデル*ABDAF*−*D*と、資源の固定費用を考慮するがアークデザイン費用を考慮しないモデル*ABDAF*−*F*を対象とする。

数値実験で使用したソフトウエア・機器等は以下の通りである。

- 使用OSおよび言語：UBUNTU 20、Python 2.7
- 最適化ソルバー：Gurobi 9.1
- CPU AMD Ryzen9 3950X 3.5GHz 16Cores、RAM 64GByte

また、数値実験で使用した設定したパラメータは以下の通りである。

- 時間パラメータ：1.2
- 配送時間制限集合：0.8、1.2
- 需要カバー率：0.5、1.0
- 最大計算時間：3時間

なお、時間パラメータはHellsten et al. (2021) で使用されている時間制限を設定する倍数である。配送時間制限の0.8は、1.2に対して2/3倍の時間制限となる。需要カバー率の1.0は配送時間制限1.2に対して全需要、0.5は配送時間制限0.8に対して全体の50%をカバーすること意味している。

使用したR問題は時間制約と容量制約をもつネットワーク設計問題のためのインスタンスである。このインスタンスを資源数の制約のないモデルに適用できるようにするために、資源の単位当たりの容量とデザイン費用を、アークの容量とデザイン費用の値を1/1、1/5、1/10 倍とした3通りのインスタンスを

生成し、それぞれをA1、A5、A10とした。A1は1資源当たりの容量が大きく、費用が高く、A10は1資源当たりの容量が小さく、費用が安いインスタンスである。需要量は同一のため、A1は使用する総資源数は相対的に少なく、A10は使用する総資源数は相対的に多くなる。*ABDAF*における最大稼働時間は、Hellsten et al.（2021）による品種中の時間制限の最大値とした。また、*ABDAF−F*おける資源の固定費用はアークのデザイン費用の平均値の30倍とした。また、*ABDAF−D*と*ABDAF−F*におけるA1、A5、A10の最大資源数をそれぞれ50、100、150とした。

4つのモデルをPythonとGurobiを用いて、最大3時間の計算時間で定式化を直接解くことにより、最適値または下界値と上界値を求めた。

表3に*IDAF*の計算結果を示す。結果はr04からr18までのグループごとに集計した。最適解数は9個のインスタンスのうち、計算時間の上限である3時間以内に最適解を求めることができたインスタンス数、近似解数は最適解を除く近似解を求めることができたインスタンス数であり、未解数は実行可能解を求めることができなかったインスタンス数である。誤差は（上界値-下界値）/下界値×100で算出し、平均誤差は実行可能解が求められたインスタンスの平均値である。また、平均時間は計算時間の平均値であり、最適解が求められない場合は3時間（10800秒）となる。

A1では、126インスタンスのうち、115インスタンスの最適解、11インスタンスの近似解を求めることができ、すべてのインスタンスで実行可能解を算出することができた。平均誤差は0.1%、最大で0.6%であり、平均計算時間は1102秒であった。A5では、121インスタンスの最適解、5インスタンスの近似解を求めることができた。平均誤差は0.0%、最大で0.2%であり、平均計算時間は511秒であった。A10では、126すべてのインスタンスの最適解を求めることができた。平均誤差は0.0%であり、平均計算時間は24秒であった。A10はA1と比べ、資源数が多くなるインスタンスであり、デザイン変数が比較的大きな整数となることから、容易な問題となっていることが分かる。また、最大で0.6%程度の誤差であるため、近似解も有用であると考えられる。

表4に*BDAF*の計算結果を示す。A1では、126インスタンスのうち、97インスタンスの最適解、8インスタンスの近似解を求めることができたが、3時間以内で21インスタンスの実行可能解を算出することができなかった。実行可能解が得られたインスタンスの平均誤差は0.4%、最大で4.9%であり、平均計算時間は1693 秒であった。A5では、92インスタンスの最適解、6インスタンスの近似解を求めることができたが、28 インスタンスの実行可能解を算出することができなかった。実行可能解が得られたインスタンスの平均誤差は0.8%、最大で6.3%と大きくなった。また、平均計算時間は2463秒であった。A10では、86インスタンスの最適解、1インスタンスの近似解を求めることができたが、39 インスタンスの実行可能解を算出することができなかった。実行可能解が得られたインスタンス

の平均誤差は0.1%、最大で0.9%、平均計算時間は1267秒であった。

*BDAF*と*IDAF*は同じ問題である。資源数の上限が大きなインスタンスでは*BDAF*の離散変数の数が膨大なものとなるため、単純に汎用の最適化ソルバーを用いて求解する場合は、整数変数を用いた方が良いことが分かる。しかしながら、さらに規模の大きなインスタンスになると、*IDAF*も解を算出することが困難になることが考えられるので、0-1変数とパスフローによる定式化に列生成法や分枝価格法などを適用した解法などを検討する必要がある。

表5に*ABDAF－D*の計算結果を示す。*ABDAF－D*は資源の固定費用を0としたものである。A1では、126インスタンスのうち、108インスタンスの最適解、18インスタンスの近似解を求めることができ、すべてのインスタンスで実行可能解を算出することができた。平均誤差は0.2%、最大で1.0%であり、平均計算時間は1804秒であった。また、平均の資源数は14.7であった。A5では、96インスタンスの最適解、26インスタンスの近似解を求めることができ、すべてのインスタンスで実行可能解を算出することができた。平均誤差は0.1%、最大で0.5%であり、平均計算時間は2641秒であった。また、平均の資源数は42.5であった。A10では、92インスタンスの最適解、27インスタンスの近似解を求めることができ、すべてのインスタンスで実行可能解を算出することができた。平均誤差は0.1%、最大で0.5%であり、平均計算時間は2786秒であった。また、平均の資源数は71.5であった。

*ABDAF－D*は*IDAF*に資源バランス条件を加えたモデルであるが、得られた結果からは問題の困難性の増加は見られない。また、最大で0.5%程度の誤差であるため、近似解も有用であると考えられる。

表6に*ABDAF－F*の計算結果を示す。*ABDAF－F*は資源の固定費用を考慮し、アークのデザイン費用を0としたものである。A1では、126インスタンスのうち、64インスタンスの最適解、61インスタンスの近似解を求めることができたが、1つのインスタンスで実行可能解を算出することができなかった。実行可能解を算出できたインスタンスの平均誤差は0.4%、最大で1.0%であり、平均計算時間は5750秒であった。また、平均の資源数は5.8であった。A5では、24インスタンスの最適解、96インスタンスの近似解を求めることができたが、6インスタンスで実行可能解を算出することができなかった。実行可能解を算出できたインスタンスの平均誤差は0.5%、最大で1.1%であり、平均計算時間は8881秒であった。また、平均の資源数は15.7 であった。多くのインスタンスグループで最適解を求めることができず、平均計算時間が上限の10800秒となっている。A10では、7インスタンスの最適解、107インスタンスの近似解を求めることができたが、12インスタンスで実行可能解を算出することができなかった。実行可能解を算出できたインスタンスの平均誤差は0.4%、最大で0.8%であり、平均計算時間は10409秒であった。また、平均の資源数は25.4であった。多くのインスタンスグループで最適解を求めることができず、平均計算時

間が上限の10800秒となっている。

*ABDAF−D*と比較すると、最適解を求めることが困難で、3時間以内で近似解を求めることも難しくなり、資源の固定費用の考慮が問題の困難性に大きな影響を与えていることが分かる。

7　おわりに

本研究では、当日配送、翌日配送などの複数の配送時間制約を考慮し、配送車である資源のバランスを考慮する資源バランス設計条件と同一品種のフローが１つのパスを流れる条件を考慮した複数の資源の配置とフローを求めるアーク設計問題のモデルを扱った。このモデルに対して、配送時間制限の考慮方法の異なるいくつかの定式化、およびフローや資源の表現が異なるいくつかの定式化を提示した。

ベンチマーク問題を用いて、4つの定式化について数値実験を行い、問題の条件や定式化による求解の困難性を明らかにした。数値実験は陽的に定式化されたものを最適化ソルバーで解くことによって行った。しかし、ベンチマーク問題においても、一定の計算時間の下で実行可能解すら算出できないインスタンスが存在した。このように、陽的に定式化を直接解くことには限界があるため、パスフローや資源サイクルの列生成法の適用、線形緩和解を用いた近似解法の開発や分枝価格法の適用など、多くの課題が残されている。

表1:R問題:属性1

問題	ノード	アーク	品種	問題	ノード	アーク	品種
r04	10	60	10	r12	20	120	200
r05	10	60	25	r13	20	220	40
r07	10	83	10	r14	20	220	100
r08	10	83	25	r15	20	220	200
r09	10	83	50	r16	20	315	40
r10	20	120	40	r17	20	315	100
r11	20	120	100	r18	20	315	200

表2:R問題:属性2

問題	容量	費用
r*-1	c1	f01
r*-2	c1	f05
r*-3	c1	f10
r*-4	c2	f01
r*-5	c2	f05
r*-7	c8	f01
r*-8	c8	f05
r*-9	c8	f10

表3:IDAFの結果

問題	費用係数	最適解数	近似解数	未解数	平均誤差 (%)	平均時間 (s)
r04	A1	9	0	0	0.0	0
r05	A1	9	0	0	0.0	0
r07	A1	9	0	0	0.0	0
r08	A1	9	0	0	0.0	0
r09	A1	9	0	0	0.0	0
r10	A1	9	0	0	0.0	1
r11	A1	9	0	0	0.0	970
r12	A1	4	5	0	0.6	6339
r13	A1	9	0	0	0.0	3
r14	A1	9	0	0	0.0	32
r15	A1	5	4	0	0.4	5575
r16	A1	9	0	0	0.0	3
r17	A1	9	0	0	0.0	17
r18	A1	7	2	0	0.3	2489
計/平均		115	11	0	0.1	1102
r04	A5	9	0	0	0.0	0
r05	A5	9	0	0	0.0	0
r07	A5	9	0	0	0.0	0
r08	A5	9	0	0	0.0	0
r09	A5	9	0	0	0.0	0
r10	A5	9	0	0	0.0	0
r11	A5	9	0	0	0.0	8
r12	A5	8	1	0	0.0	1366
r13	A5	9	0	0	0.0	0
r14	A5	9	0	0	0.0	5
r15	A5	6	3	0	0.2	4099
r16	A5	9	0	0	0.0	1
r17	A5	9	0	0	0.0	4
r18	A5	8	1	0	0.0	1671
計/平均		121	5	0	0.0	511
r04	A10	9	0	0	0.0	0
r05	A10	9	0	0	0.0	0
r07	A10	9	0	0	0.0	0
r08	A10	9	0	0	0.0	0
r09	A10	9	0	0	0.0	0
r10	A10	9	0	0	0.0	0
r11	A10	9	0	0	0.0	1
r12	A10	9	0	0	0.6	6
r13	A10	9	0	0	0.0	0
r14	A10	9	0	0	0.0	1
r15	A10	9	0	0	0.0	304
r16	A10	9	0	0	0.0	0
r17	A10	9	0	0	0.0	1
r18	A10	9	0	0	0.0	15
計/平均		126	0	0	0.0	24

表4:BDAFの結果

問題	費用係数	最適解数	近似解数	未解数	平均誤差 (%)	平均時間 (s)
r04	A1	9	0	0	0.0	0
r05	A1	9	0	0	0.0	3
r07	A1	9	0	0	0.0	0
r08	A1	9	0	0	0.0	2
r09	A1	9	0	0	0.0	16
r10	A1	9	0	0	0.0	49
r11	A1	3	4	2	1.1	7424
r12	A1	0	4	5	4.9	1080
r13	A1	9	0	0	0.0	50
r14	A1	6	0	3	0.0	63
r15	A1	3	0	6	0.0	4496
r16	A1	9	0	0	0.0	37
r17	A1	7	0	2	0.0	90
r18	A1	6	0	3	0.0	667
計/平均		97	8	21	0.4	1693
r04	A5	9	0	0	0.0	1
r05	A5	9	0	0	0.0	15
r07	A5	9	0	0	0.0	2
r08	A5	9	0	0	0.0	13
r09	A5	9	0	0	0.0	73
r10	A5	9	0	0	0.0	280
r11	A5	4	2	3	0.4	6685
r12	A5	0	2	7	2.3	10801
r13	A5	9	0	0	0.0	176
r14	A5	6	0	3	0.0	1940
r15	A5	1	1	7	6.3	7768
r16	A5	9	0	0	0.0	111
r17	A5	6	0	3	0.0	541
r18	A5	3	1	5	1.5	6070
計/平均		92	6	28	0.8	2463
r04	A10	9	0	0	0.0	3
r05	A10	9	0	0	0.0	46
r07	A10	9	0	0	0.0	4
r08	A10	9	0	0	0.0	39
r09	A10	9	0	0	0.0	202
r10	A10	9	0	0	0.0	817
r11	A10	3	1	5	0.9	6516
r12	A10	0	0	9	0.0	0
r13	A10	9	0	0	0.0	334
r14	A10	4	0	5	0.0	944
r15	A10	1	0	8	0.0	6569
r16	A10	9	0	0	0.0	276
r17	A10	5	0	4	0.0	486
r18	A10	1	0	8	0.0	1504
計/平均		86	1	39	0.1	1267

表5:ABDAF−Dの結果

問題	費用係数	最適解数	近似解数	未解数	平均誤差 (%)	平均時間 (s)	平均資源数
r04	A1	9	0	0	0.0	0	8.1
r05	A1	9	0	0	0.0	2	9.6
r07	A1	9	0	0	0.0	0	6.6
r08	A1	9	0	0	0.0	1	10.0
r09	A1	9	0	0	0.0	5	13.3
r10	A1	9	0	0	0.0	23	11.7
r11	A1	8	1	0	0.1	2470	14.9
r12	A1	1	8	0	1.0	9790	23.9
r13	A1	9	0	0	0.0	32	12.2
r14	A1	8	1	0	0.0	2049	18.7
r15	A1	4	5	0	1.0	6585	20.4
r16	A1	9	0	0	0.0	94	10.1
r17	A1	9	0	0	0.0	453	20.9
r18	A1	6	3	0	0.6	3748	25.3
計/平均		108	18	0	0.2	1804	14.7
r04	A5	9	0	0	0.0	0	21.7
r05	A5	9	0	0	0.0	12	36.2
r07	A5	9	0	0	0.0	4	23.8
r08	A5	9	0	0	0.0	5	32.3
r09	A5	8	1	0	0.0	1207	38.9
r10	A5	7	2	0	0.0	2428	40.7
r11	A5	6	3	0	0.1	4572	46.6
r12	A5	2	7	0	0.5	8966	71.2
r13	A5	9	0	0	0.0	133	35.9
r14	A5	6	2	0	0.0	3036	50.8
r15	A5	3	6	0	0.5	8127	63.6
r16	A5	9	0	0	0.0	86	28.6
r17	A5	8	0	0	0.0	553	50.9
r18	A5	2	5	0	0.4	7850	54.1
計/平均		96	26	0	0.1	2641	42.5
r04	A10	9	0	0	0.0	1	39.4
r05	A10	9	0	0	0.0	13	62.1
r07	A10	9	0	0	0.0	5	45.3
r08	A10	9	0	0	0.0	4	58.4
r09	A10	9	0	0	0.0	38	69.7
r10	A10	7	2	0	0.0	3017	71.4
r11	A10	6	3	0	0.1	4554	89.2
r12	A10	1	8	0	0.5	9672	113.4
r13	A10	9	0	0	0.0	263	61.3
r14	A10	7	1	0	0.0	2582	87.3
r15	A10	1	8	0	0.2	9633	104.2
r16	A10	8	1	0	0.0	1324	51.4
r17	A10	6	0	0	0.0	318	64.0
r18	A10	2	4	0	0.1	7585	83.0
計/平均		92	27	0	0.1	2786	71.5

表6:ABDAF−Fの結果

問題	費用係数	最適解数	近似解数	未解数	平均誤差 (%)	平均時間 (s)	平均資源数
r04	A1	9	0	0	0.0	29	2.9
r05	A1	9	0	0	0.0	102	2.6
r07	A1	9	0	0	0.0	48	2.8
r08	A1	8	1	0	0.1	1468	3.3
r09	A1	7	2	0	0.0	3390	4.1
r10	A1	7	2	0	0.0	3932	3.2
r11	A1	2	7	0	0.8	8939	3.4
r12	A1	0	9	0	1.0	10800	4.3
r13	A1	6	3	0	0.1	4871	4.3
r14	A1	0	9	0	0.8	10800	10.6
r15	A1	1	7	1	0.9	10247	9.0
r16	A1	6	3	0	0.0	4273	3.3
r17	A1	0	9	0	0.8	10800	13.1
r18	A1	0	9	0	1.0	10800	14.6
計/平均		64	61	1	0.4	5750	5.8
r04	A5	7	2	0	0.1	2656	9.6
r05	A5	5	4	0	0.1	4983	9.2
r07	A5	5	4	0	0.2	5092	8.9
r08	A5	3	6	0	0.2	7230	10.4
r09	A5	2	7	0	0.1	9226	11.8
r10	A5	2	7	0	0.3	8739	11.3
r11	A5	0	9	0	0.5	10800	11.3
r12	A5	0	9	0	0.5	10800	15.7
r13	A5	0	9	0	0.4	10800	13.7
r14	A5	0	9	0	1.0	10800	35.4
r15	A5	0	9	0	1.1	10800	28.9
r16	A5	0	9	0	0.4	10800	10.7
r17	A5	0	6	3	0.6	10800	19.2
r18	A5	0	6	3	0.9	10800	23.8
計/平均		24	96	6	0.5	8881	15.7
r04	A10	3	6	0	0.2	7827	16.7
r05	A10	1	8	0	0.1	9697	16.2
r07	A10	3	6	0	0.2	9404	16.8
r08	A10	0	9	0	0.2	10800	18.3
r09	A10	0	9	0	0.2	10800	21.6
r10	A10	0	9	0	0.5	10800	21.1
r11	A10	0	9	0	0.6	10800	22.0
r12	A10	0	9	0	0.6	10800	30.7
r13	A10	0	9	0	0.4	10800	24.7
r14	A10	0	6	3	0.6	10800	31.7
r15	A10	0	6	3	0.6	10800	40.3
r16	A10	0	9	0	0.3	10800	18.2
r17	A10	0	6	3	0.5	10800	33.5
r18	A10	0	6	3	0.8	10800	43.3
計/平均		7	107	12	0.4	10409	25.4

参考文献

Andersen, J., M. Christiansen, T. G. Crainic, R. Grønhaug. 2011. Branch and price for service network design with asset management constraints. *Transportation Science* 45(1) 33–49.

Andersen, J., T. G. Crainic, M. Christiansen. 2009. Service network design with management and coordination of multiple fleets. *European Journal of Operational Rsearch* 193(2) 377–389.

Atamtürk, A., D. Rajan. 2002. On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Mathematical Programming* 92(2) 315–333.

Bai, R., S. W. Wallace, J. Li, A. Y. Chong. 2014. Stochastic service network design with rerouting. *Transportation Research Part B* 60 50–65.

Balakrishnan, A., C.V. Karsten. 2017. Container shipping service selection and cargo routing with transshipment limits. *European Journal of Operational Research* (263(2)) 652—-663.

Boland, N., M. Hewitt, L. Marshall, M. Savelsbergh. 2017. The continuous-time service network design problem. *Operations Research* 65 1115–1428.

Crainic, T. G., A. Frangioni, B. Gendron. 2001. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design problems. *Discrete Applied Mathematics* 112(1-3) 73–99.

Crainic, T. G., Fu X, M. Gendreau, W. Rei, S. Wallace. 2011. Progressive hedgingbased metaheuristics for stochastic network design. *Networks* 58 114–124.

Crainic, T.G., M. Gendreau, B.Gendron, eds. 2021. *Network Design with Applications to Transportation and Logistics. Springer.*

Crainic, T.G., M. Hewitt, M. Toulouse, D.M. Vu. 2016. Service network design with resource constraints. *Transportation Science* 50(4) 1380–1393.

Croxton, K. L., B. Gendron, T. L. Magnanti. 2003. A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Management Science* 49 1268–1273.

Erera, A., M. Hewitt, M. Savelsbergh, Y. Zhang. 2013. Improved load plan design through integer programming based local search. *Transportation Science* 47(3) 295–454.

Hellsten, E., D.F. Koza, I. Contreras, J.F. Cordeau, D. Pisinger. 2021. The transit time constrained fixed charge multi-commodity network design problem. *Computers & Operations Research* 136 105511.

Jarrah, A. I., E. Johnson, L. C. Neubert. 2009. Large-scale, less-than-truckload service network design. *Operations Research* 57(3) 609–625.

Karsten, C.V., B.D. Brouer, D. Pisinger. 2017. Integrated liner shipping network design and scheduling. *Transportation Research Part E: Logistics and Transportation Review* (105) 152—-162.

Koza, D.F., G. Desaulniers, S. Ropke. 2020. Integrated liner shipping network design and scheduling. *Transportation Science* (Published Online).

Li, X., K. Wei, Y.P. Aneja, P. Tian, Y. Cui. 2017. Matheuristics for the single-path design-balanced service network design problem. *Computers & Operations Research* 77 141–153.

Pedersen, M. B., T. G. Crainic, O. B. G. Madsen. 2009. Models and tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science* 43 158–177.

Rahmaniani, R., T. G. Crainic, M. Gendreau, W. Rei. 2018. Accelerating the benders decomposition method: Application to stochastic network design problems. *SIAM Journal on Optimization* 28(1) 875–903.

Tierney, K., J.F. Ehmke, A.M. Campbell, D. Müller. 2019. Liner shipping single service design problem with arrival time service levels. *Flexible Services and Manufacturing Journal* (31(3)) 620—-652.

Trivella, A., F. Corman, D. F. Koza, D. Pisinger. 2021. The multi-commodity network flow problem with soft transit time constraints: Application to liner shipping. *Transportation Research Part E: Logistics and Transportation Review* 150 102342.

Wang, X., T. G. Crainic, S. W.Wallace. 2019. Stochastic network design for planning scheduled transportation services: the value of deterministic solutions. *Informs Journal on Computing* 31. doi:10.1287/ijoc.2018.0819.

Yaghini, M., M. R. A Kazemzadeh. 2012. A simulated annealing algorithm for unsplittable capacitated network design. *International Journal of Industrial Engineering & Production Research* 23(2) 91–100.