モデル検査法によるドローン物流システムの近似スケジューリング

清水 圭太　長谷部 浩二

近年，物流業界におけるドライバー不足などの問題を解決する方法として，ドローン（無人航空機）を用いた配送システムが注目されている．ドローンの運用においては，航続距離や衝突の回避などの制約条件を満たす輸送効率の高いスケジューリングが求められる．このような交通輸送システムのスケジュールを生成するために，モデル検査法を用いた手法が存在する．しかし，モデル検査法には，対象とするシステムの規模が大きくなると，しばしば状態爆発を引き起こすという問題がある．そこで本研究では，飛行区域と時間の両方を抽象化するスケジューリング手法を提案する．具体的には，飛行区域の粒度を変更し，それに応じて単位時間の粒度も変更することでモデルを抽象化する．さらに，抽象化して得られたスケジュールをもとに，それぞれの区域で詳細化を行うというものである．また本研究では，具体例をもとに提案手法の有用性を示す．

1　序論

モデル検査法 [6] は，システムの仕様が安全性などの性質を満たしているかどうかを自動的に検証するための形式手法の一種である．モデル検査法では，対象とするシステムの振る舞いを状態遷移モデルとして記述し，システムが満たすべき性質を線型時間論理（Linear Temporal Logic：LTL）や計算木論理（Computational Tree Logic：CTL）などの時相論理式で記述する．これらの入力から，システムがとりうる状態全てを網羅的に調べることで，システムが性質を満たすかどうかを検証する．システムが性質を満たす場合には「valid」を出力し，そうでない場合には初期状態から性質が満たされない状態に至るまでの実行列である反例を出力する．モデル検査法はさまざまなソフトウェアやハードウェアの設計に広く利用されており，SPIN [8] や NuSMV [5] など多くのモデル検査器が存在する．

モデル検査法はスケジューリングなどの離散最適化問題にも適用することができる．その基本的なアイディアは最適化問題を到達可能性問題へと還元するというものである．すなわち，対象とするシステムのモデルに対して，スケジュールが満たすべき条件を記述した論理式の否定を入力として与えて検証を行い，その検証によって得られた反例をスケジュールとして解釈するというものである．モデル検査法を用いたスケジューリングの例としては，Behrmann らの研究 [1] が挙げられる．そこでは時間オートマトンを扱うことができるモデル検査器である UPPAAL [2] を用いて，さまざまなスケジューリング問題の求解を行なっている．また，UPPAAL を用いた例としては，製鉄所の製造工程のスケジューリングに関する Fehnker の研究 [7] が挙げられる．

モデル検査法を用いたスケジューリング手法には，作成したモデルを再利用して，システムの安全性などの性質を検証できるという利点がある．一方で，モデル検査法には状態爆発が起こりやすいという課題も存在する．スケジューリングにおける状態爆発を回

Approximate Scheduling for Drone Delivery System Using Model Checking

Keita Shimizu, 筑波大学システム情報学群コンピュータサイエンス専攻, Department of Computer Science, University of Tsukuba.

Koji Hasebe, 筑波大学システム情報系, Faculty of Engineering, Information and Systems, University of Tsukuba.

避するためには主に 2 つのアプローチが考えられる．1 つ目は，モデル検査法が最適化問題を解く際に必ずしも効率的な探索を行わないことから，モデルを改良して探索を効率よく行うようにするというものである．そのような例として，Ruys [9] による分枝限定法を取り入れたモデルの作成方法がある．しかし，こうした方法によってモデルを作成するのは一般に容易ではなく，探索アルゴリズムに精通していなければならないという問題がある．またもう 1 つの方法としては，モデルを抽象化することによって状態数を減らすというものがある．状態爆発を回避することを目的とした抽象化の技術は，類似する複数の状態を 1 つの状態にまとめる方法など数多く存在するが，スケジューリングに適した抽象化については未だ十分検討されていない．

そこで本論文では，モデル検査法を用いた近似スケジューリング手法を提案する．特にここでは，スケジューリングの対象をドローンを用いた物流システムとし，ドローンのバッテリ残量や衝突の回避などの制約を満たす最短のスケジュールの生成を目的とする．本研究で提案する近似化の基本的なアイディアは，2 次元格子空間によってモデル化されたドローンの飛行エリアに対して，複数のマス目を 1 つの大きなマス目にまとめるという抽象化を行うというものである．その上で，抽象化によって得られたスケジュールをもとに，抽象化したマス目の内部におけるスケジュールの詳細化を行うことにより，全体の具体的なスケジュールを得る．以上で述べた手法の利点としては，組み合わせるマスの数を調整することで任意の抽象化レベルを選択できる点が挙げられる．また，抽象化のためのモデルの変更が単純であるということも，利点として挙げられる．

本論文では，モデル検査器として，大規模で特に同期プロセスによるシステムの記述に適した NuSMV [5] の後継である nuXmv [4] を用いた．さらに，制約条件を満たす最短スケジュールを得るために，nuXmv が提供する有界モデル検査 [3] を用いて検証を行なった．その結果，例えば厳密解となるスケジュールが 1 時間以内で求めることができなかった 4×4 で区切られた飛行エリアの場合でも，抽象化を行うことでスケジュールを 5 分以内に得ることができた．

本論文の構成は以下の通りである．第 2 章では，対象とするシステムの概要を述べる．第 3 章では，そのモデル化の方法について説明する．第 4 章では，提案手法の 1 つである抽象化の方法について説明する．第 5 章では抽象化によって得られた解を具体化する方法について説明する．最後に，第 6 章で結論と今後の研究課題について述べる．

2　ドローンによる物流システムの概要

この章では，対象とするドローンによる物流システムの概要について説明する．

2.1　飛行エリア

このシステムでは，あらかじめドローンが飛行できるエリアが定められている．これを飛行エリアと呼ぶ．このシステムの目的は，デポにある荷物を飛行エリア内の指定された場所に配送することである．飛行エリアの例を図 1 に示す．飛行エリアは格子状に区切られており，各マスはドローンの充電や離着陸，荷物の搭載などを行う 1 つ以上のデポとそれ以外のマスに分類できる．本論文では，デポ以外のマスを通常マスと呼ぶ．通常マスに書かれている数字はその通常マスにおける顧客の需要を表している．各需要は荷物を持つドローンがその通常マスを飛行して荷物を配送した際に，その値を 1 減らすことができる．

	A	B	C	D	
1	1	0	1	0	
2	0	1	0	0	
3	1	0	1	1	
4	0	0	0	0	depot

図 1　飛行エリアの例

2.2 ドローン

この物流システムで扱うドローンは，内蔵されているバッテリによって 1 回の充電で飛行できる時間が限られている．しかし，デポで充電を行うことにより飛行する時間を増やすことが可能である．対象の物流システムでは，複数のドローンがすべての通常マスにおける需要を 0 にするために飛行を繰り返す．各マスにいるドローンは，自身が現在いるマスと縦もしくは横に隣接するマスに移動するか，同じマスにとどまるホバリングをするものとする．なお本論文では，1 単位時間をドローンがあるマスから隣接しているマスへ移動するのに必要な時間とし，すべてのマスで移動するために消費する時間は同じものとする．

また，スケジュールによってはある時刻に複数のドローンが狭い領域内に存在することによって衝突が起きてしまう可能性がある．そこで本論文では，同じ時刻に同じ通常マスに滞在することができるドローン数の最大数を設けることでこの衝突を回避させるものとする．

3 対象とする物流システムのモデル化

この章では，ドローンのスケジューリングのためのモデル化について述べる．

3.1 飛行エリアのモデル化

飛行エリア F は m 行 n 列の 2 次元格子空間からなり，通常マスとデポと呼ばれる特別なマス目から構成される．ここで，i 行 j 列のマス目を (i, j) と表記する．なお，本論文では一般性を失うことなく議論を単純化するために，先の図 1 で示すように，デポを 1 つだけ格子空間の外側で隣接するマス目として配置することとする．ここで，デポの集合を $Depot$，通常マスの集合を C とし，$F = Depot \cup C$，$Depot \cap C = \emptyset$ であるとする．また，通常マスには客の需要が存在し，これを表す関数を $needs\colon C \to \mathbb{N}$ とする．さらに，ドローン同士の衝突を避けるための各通常マスに同時に滞在可能なドローンの最大数を表す関数を $cap\colon C \to \mathbb{N}$ とする．

3.2 ドローンの動作のモデル化

ドローン全体の集合を D とする．各ドローンはバッテリが許す飛行時間以内でデポを離陸し，需要が 1 以上の通常マスに荷物を配達したのちにデポに着陸する必要がある．その飛行時間の上限を表す関数を $max\_fly\colon D \to \mathbb{N}$ とする．

モデルにおける時刻を，$0, 1, 2, \ldots \in T(= \{\mathbb{N}\})$ とする．ドローンのスケジュールは，各時刻におけるそれぞれのドローンの飛行エリアの位置を定めることによって定義される．すなわち，長さ h のドローンのスケジュールとは，関数 $sche\colon T_h \times D \to F$ のことであるとする．（ただし，$T_h = \{t \mid 0 \leq t \leq h\}$ とする．）ここで，各ドローンはデポにて離着陸をするため，$sche(0, d), sche(h, d) \in Depot$ を満たすものとする．また，時刻 $t-1$ から時刻 t へのドローンの移動は縦横のマスへの移動もしくはホバリングの 3 種類のいずれかである．したがって，任意の $t \in T_h$ と任意の $d \in D$ について，$sche(t, d) = (i, j)$ かつ $sche(t+1, d) = (i'j')$ ならば，$i = i'$ かつ $|j - j'| \leq 1$ または $|i - i'| \leq 1$ かつ $j = j'$ を満たすものとする．

4 モデル検査によるスケジューリング手法

この章では，第 3 章でモデル化したシステムをスケジューリングする方法について説明する．さらに，対象とするシステムを抽象化することによる近似スケジューリング方法について説明する．

4.1 モデル検査法によるスケジューリング

まず，前節で述べたモデルを NuSMV の後継の nuXmv の仕様記述言語である SMV で記述する．次にスケジュールが満たすべき条件の連言の否定を表す論理式をモデル検査法で検証し，反例が出るかを確認する．もし，反例が生成された場合，その反例は与えられた条件の否定の否定，すなわち条件を満たした実行過程であり，求めるべきスケジュールとみなすことができる．また，本研究では検証方法として nuXmv の有界モデル検査を用いる．有界モデル検査は初期状態から決められた遷移数までのモデル検査を行う手法である．また，遷移数が少ない方から検証を行い，反例が出ると同時に検証を終了するため，最も短いス

ケジュールのうちの 1 つを得ることができる．

4.2 スケジュールが満たすべき条件

この節では，システムがスケジュールを生成する際に，モデル検査法に与える制約について説明する．この制約はスケジュールの終了条件と物流システムそのものの制約の 2 種類に分類される．また本研究では，スケジュールとなる実行パスについて検証を行うため，時相論理式として LTL を用いる．

(F-1) 「いつかすべてのドローンがいずれかのデポにいる．」

ここで，現在のドローンの位置を表す変数を $pos_d \in F$ とする．これを用いると LTL 式は以下のようになる．なお，$\Diamond$ は内側の論理式が未来のある時点で成り立つときに真になるような様相演算記号である．

$$\Diamond \bigwedge_{d \in D} (pos_d \in Depot)$$

(F-2) 「すべての通常マスについて，いつか需要が 0 になる．」

ここで，現在通常マスに止まっている荷物を持つドローンの台数を表す変数を $hasBag_c \in \{0, 1, \cdots, |D|\}$ とする．また，現在の通常マスの需要を表す変数を $nowneeds_c \in \{0, 1, \cdots, needs(c)\}$ とする．この $nowneeds$ は $0 \leq nowneeds \leq needs$ を取るものとし，$hasBag$ の値によって非決定的に変化する．

$$nowneeds_c = \begin{cases} nowneeds_c - hasBag_c \\ nowneeds_c \end{cases}$$

この $nowneeds_c$ を用いると，LTL 式は以下のように表すことができる．

$$\Diamond \bigwedge_{c \in C} (nowneeds_c \leq cap(c))$$

(S-1) 「すべてのドローンについて，デポにいる，もしくは通常マスにいるときのバッテリーの残量が 0 以上である．」

ここで，現在のドローンのバッテリー残量で移動できる時間表す変数を $flytime_d \in \{0, 1, \cdots, max\_fly(d)\}$ とする．この $flytime$ はドローンが現在いるマスの種類によって以下のように値が変化する．

$$flytime_d = \begin{cases} max\_fly(d) & (pos_d \in Depot) \\ flytime'_d - 1 & (otherwise) \end{cases}$$

ただしここで $flytime'_d$ は現在の 1 つ前のステップにおけるドローンのバッテリー残量を表すものとする．この $flytime_d$ を用いると，LTL 式は以下のように表すことができる．なお，$\Box$ は内側の論理式が常に成り立つときに真になるような様相演算記号である．

$$\bigwedge_{d \in D} \Box(pos_d \in Depot \vee flytime_d > 0)$$

(S-2) 「すべての通常マスについて，通常マス内に滞在しているドローンの台数はその通常マスの最大数以下である．」

ここで，現在通常マスに止まっているドローンの台数を $park_c \in \{0, 1, \cdots, cap(c)\}$ とする．これと 3.1 節で述べた通常マスに滞在できるドローンの最大数を表す関数の cap を用いると，LTL 式は以下のように表すことができる．

$$\bigwedge_{c \in C} \Box(park_c \leq cap(c))$$

以下，本論文では，すべての通常マスの $cap = 1$ とし，すべてのドローンの $max\_fly = 30$ とする．さらに，対象の物流システムのモデルに対して，上記の 4 項目の論理積の否定の式をモデル検査法にて検証すると，スケジュールを表す反例を得ることができる．

4.3 モデルの抽象化

この節では，ドローンのスケジュールの近似解を求めるために，提案手法の 1 つである飛行エリアの抽象化について説明する．本研究では，複数の通常マスを 1 つの大きな通常マスにまとめる抽象化を行うことで，スケジュールの近似解を求める手法を提案す

る．ここで，$k_m, k_n \in \mathbb{N}(>1)$ とする．また，m 行 n 列の矩形で表されていた飛行エリアの通常マスの部分について，縦が k_m 個，横が k_n 個の通常マスから構成される新たな通常マス $c_{app} \in C_{app}, C_{app} \subseteq C$ を生成することで，m/k_m 行 n/k_n 列の新たな飛行エリアに抽象化する．この抽象化した通常マス c_{app} を抽象通常マスと呼ぶことにする．なお，デポは通常マスと変数等の扱いが異なるため結合しないものとする．図 2 は図 1 の飛行エリアを $k_m = 2$，$k_n = 2$ で抽象化したときの新しい飛行エリアである．

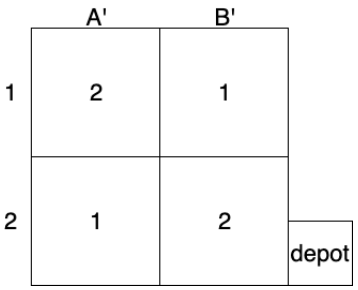

図 2　図 1 で示した飛行エリアを $k_m = 2$，$k_n = 2$ で抽象化したときの飛行エリア

次に，抽象化によって変更が必要になる関数について説明する．ここで，抽象通常マスに含まれる元の通常マスを表す関数を $cons\colon C_{app} \to 2^C$ とする．まず，3.1 節で述べた需要 $needs$ について，抽象化した状態における新しい需要を $needs_{app}\colon C_{app} \to \mathbb{N}$ は，$needs_{app}(c_{app}) = \sum_{c \in cons(c_{app})} needs(c)$ とすることができる．次に，各通常マスにおけるドローンの最大数 cap について，抽象通常マスにおけるドローンの最大数 $cap_{app}\colon C_{app} \to \mathbb{N}$ は，$cap_{app} = \sum_{c \in cons(c_{app}))} cap(c)$ となる．最後に，3.2 節で述べたドローンの飛行時間の上限 max_fly が抽象化によって変更されたときの値を $max\_fly_{app}\colon D \to \mathbb{N}$ とすると，$max\_fly_{app}(d) = \lfloor max\_fly(d)/(2 * max(k_m, k_n) + min(k_m, k_n) - 1) \rfloor$ となる．これは，元の飛行時間の上限を抽象通常マスにおいて同じ辺から入って同じ辺から出るまでにかかる最大の飛行時間で割ったものである．このとき，時刻の粒度として，$k_t = 2 * max(k_m, k_n) + min(k_m, k_n) - 1$ とする．

さらに，抽象化に伴って増える 3 種類のシステムの制約について説明する．まず 1 つ目として，抽象通常マスの 1 辺から出るドローンの台数が k_m または k_n 台以下にする必要がある．2 つ目として，抽象通常マスの隣接する 2 辺から出るドローンの合計台数について，$k_m + k_n - 1$ 台以下にする必要がある．3 つ目は，抽象通常マスの隣接する 2 辺へ入るドローンの合計台数について，$k_m + k_n - 1$ 台以下にする必要がある．

最後に，抽象化したモデルに対して，これらの抽象化した値を第 4.2 節における検証式で元の値の代わりに置き換えたものの論理積の否定の式をモデル検査法にて検証することで，スケジュールの近似解を生成することができる．例えば，図 1 を抽象化した図 2 をスケジューリングを行うと図 3 のような近似解を得ることができた．図 3 では，各ドローンが通常マスにて配達したときは緑色で塗りつぶしてある．

	d_1	d_2	d_3	d_4	d_5
0	depot	depot	depot	depot	depot
1	B'2	B'2	B'2	B'2	
2		A'2	B'1	A'2	B2
3	A'2	B'2	A'1	A'1	B'1
4	B'2	depot	B'1	A'2	
5	depot	B'2	B'2	B'2	B'2
6	depot	depot	depot	depot	depot

図 3　図 2 のスケジュールを具体化した結果

5　近似解の具体化

第 4 章で説明した方法だけでは，抽象通常マス内部のスケジュールまではわからない．そこでこの章では，抽象通常マス内部をスケジューリングする方法について説明する．また本論文では，スケジュール生成時間を短縮するために，複数の通常マス内部のスケジューリングを並行して行う．そのため，隣接している抽象通常マス同士の内部のスケジュールの整合性が合わない場合が存在する，そこで，抽象通常マス内部のスケジュールの整合性を合わせることで，スケジュールを完成させる方法についても説明する．

5.1 抽象通常マス内部のスケジューリング

この節では，抽象通常マスの内部のスケジューリングを行う方法について説明する，．最初に 4 章で得られた抽象化状態のスケジュールから得ることができる情報について説明する．図 4 は図 2 における B2 マスを表す．ここで，抽象通常マス c_{app} に含まれない抽象通常マスの集合を $Out = C - \{c_{app}\}$ と呼ぶことにする．抽象化状態のスケジュールから，スケジューリング対象の抽象通常マスと $out \in Out$ または $depot \in Depot$ を出入りしたときの時刻がわかる．例えばドローン D1 の B2 への出入りは図 3 より，時刻 1 に depot から入り，時刻 2 に A'2 へ出て，時刻 5 に A'2 から入り，時刻 6 に depot へ出ることが分かる．

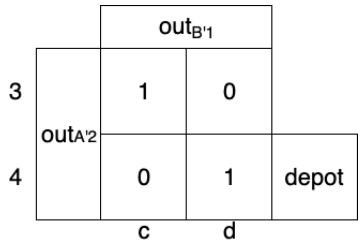

図 4　図 2 で示した B′2 の内部および周辺のマス

次に，抽象化した状態のスケジュールから抽象通常マスの内部をスケジューリングするために，抽象化した時刻におけるスケジュールから具体化した時刻におけるスケジュールへの変換について説明する．ここで，抽象化状態のスケジュールの時刻の集合を $T_{app} = \{0, \cdots, h_{app}\}$ とする．抽象化状態の時刻 $t_{app} \in T_{app}$ で表されたスケジュールを，時刻 $t \in T$ で表されたスケジュールに変換するには以下のようにする．

1. $t_{app} = 0$ のとき，$t = 0$ のスケジュールに変換する．
2. $t_{app} = 1$ のとき，$t = 1$ から $t = k_t - 1$ のスケジュールに変換する．
3. $1 < t_{app} < h_app$ のとき，$k_t * (t_{app} - 1)$ から $k_t * (t_{app}) - 1$ のスケジュールに変換する．
4. $t_{app} = h_{app}$ のとき，$t = k_t * (h_{app} - 1)$ のスケジュールに変換する．

最後に，抽象通常マスの内部をスケジュールする際に，モデル検査器に与える LTL 式について説明する．**(F-2)**，**(S-2)** はそのまま使うことができる．また，**(F-1)** はスケジューリング対象の抽象通常マスがデポに隣接していない場合には，**(F-1)** の一部を変更する必要がある．具体的には，スケジュール対象の抽象通常マスにおけるドローンの位置の最終条件として適切な out もしくは $depot$ を返す関数を $finOut\colon C_{app} \times D \rightarrow Out \cup Depot$ とする．これにより，ドローン d は適切な $out^* \in Out$ もしくは $depot$ にいるときにスケジュールが終了するようにする．これを **(F-1)'** とする．さらに，**(S-1)** は近似スケジュールが生成された時点で必ず満たすので検証する必要はない．しかし，対象の抽象通常マスとその他のマスの出入りの制約を満たすために新たな論理式を追加する必要がある．これを **(E-1)** とする．

各抽象通常マスにて，**(F-1)'**, **(F-2)**, **(S-1)**, **(E-1)** の 4 つ LTL 式の論理積の否定を入力として検証を行い，それらを結合したものを図 5 に示す．具体化した時刻において，抽象化状態の時刻 t_{app} が同じである部分は枠線の太さを変更している．さらに，赤字で示した部分は，各抽象通常マスの内部を並列にスケジューリングしたために，直前の時刻と現在の時刻でドローンの位置に整合性が取れていない部分を表している．

5.2 抽象通常マス同士の整合性のとりかた

この節では，並列にスケジューリングして組み合わせたスケジュールの整合性を保つために，ドローンの位置を移動させる方法について説明する．整合性を保つために，スケジュールのはじめから，すべてのドローンについて以下のルールを上から適用するものとする．

(R-1) デポと隣接している通常マスでホバリングしているならば，デポに移動する．

(R-2) 同じ抽象通常マスの同じ辺から入る別のドローンが存在し，かつそのドローンも整合性が取れていないならば，両者の位置を入れ替える．

	d_1	d_2	d_3	d_4	d_5
0	depot	depot	depot	depot	depot
1			D4	depot	
2			C4	D4	
3		D4	C3	C4	
4	D4	C4	D3	C3	
5	C4	B4	C2	B3	D4
6			C1	A3	D3
7					
8					
9					
10	B3	C3	B1	A2	C2
11	A3	D3	B2	A1	C1
12			B1		
13		D4			
14		depot	B2	A2	
15	C4		C2	B3	D1
16	D4		C1	A3	
17	depot				
18					
19			C2	B3	D2
20		D4	D3	C4	C3
21		D3	D4	C3	C4
22		D4	depot	D3	
23		depot		D4	
24				depot	D4
25	depot	depot	depot	depot	depot

図 5　図 3 のスケジュールを具体化したスケジュール

	d_1	d_2	d_3	d_4	d_5
0	depot	depot	depot	depot	depot
1			D4	depot	
2			C4	D4	
3		D4	C3	C4	
4	D4	C4	D3	C3	
5	C4	B4	D2	B3	D4
6			C2	A3	D3
7			C1		
8					
9					
10	B4	C4	B1	A2	D2
11	B3	C3	B2	A1	C2
12	A3	D3	B1		C1
13		D4			
14	B3	depot	B2	A2	
15	C3		C2	A3	D1
16	C4		C1		
17	D4				
18	depot			B3	
19			C2	B4	D2
20		D4	C3	C4	D3
21		D3	C4	C3	D4
22		D4		D3	depot
23		depot		D4	
24			D4	depot	
25	depot	depot	depot	depot	depot

図 6　図 5 のスケジュールを整合性が保たれるように調整したスケジュール

(R-3) 隣接している通常マスに移動させる．

(R-4) **(R-1)**，**(R-2)**，**(R-3)** によって移動させたあとのスケジュールの整合性を保つために，ドローンを移動させる．

もし，上記のルールでも整合性を保てなかった場合には，整合性を保てなかったステップの間に整合性を保てなかったドローンのみを移動させるステップを挟むことで，整合性を保つ．上記のルールに従って整合性を保つように調整したスケジュールを図 6 に示す．**(R-2)** を適用した例としては，時刻 20 におけるドローン 3 とドローン 5 の交換が挙げられる．また，**(R-3)** を適用した例としては，時刻 5 におけるドローン 3 や時刻 10 におけるドローン 1，ドローン 2，ドローン 5 などが挙げられる．

5.3　冗長な時間の削除

抽象化した状態のスケジュールを具体化しただけでは，すべてのドローンがホバリングすることによる冗長な時間な時間が発生することがある．そこで，，整合性を保たせたスケジュールに対して，冗長な部分を削除する方法について説明する．その方法は，抽象化状態の 1 ステップごとに，各抽象通常マスにいるすべてのドローンがホバリングしている時間を取り，その最小値の分だけ各ドローンのホバリングしている部分を削除するというものである．ただし，ホバリングしている時間を削除し，その結果ドローンが別の通常マスに移動することにより，各通常マスのドローンの台数が最大数に到達する場合には，削除しようとした長さから 1 引いた値で再度削除を試み，0 になった場合には削除は行わない．この削除によって，図 6

の冗長な部分を削除したスケジュールを図 7 に示す．削除した例としては，時刻 5 から時刻 9 ではすべてのドローンは少なくとも 2 単位時間分ホバリングしているので，スケジュールの長さを 2 単位時間分削除できる．冗長な部分を削除した結果，厳密解とのスケジュールの長さの差は 7 となっている．

	d_1	d_2	d_3	d_4	d_5
0	depot	depot	depot	depot	depot
1			D4		
2			C4	D4	
3		D4	C3	C4	
4	D4	C4	D3	C3	
5	C4	B4	D2	B3	D4
6			C2	A3	D3
7			C1		
8	B4	C4	B1	A2	D2
9	B3	C3	B2	A1	C2
10	A3	D3	B1		C1
11		D4			
12	B3	depot	B2	A2	
13	C3		C2	A3	D1
14	C4		C1		
15	D4			B3	
16	depot		C2	B4	D2
17		D4	C3	C4	D3
18		D3	C4	C3	D4
19		D4		D3	depot
20		depot		D4	
21			D4	depot	
22	depot	depot	depot	depot	depot

図 7　図 6 のスケジュールから冗長な部分を削除したスケジュール

6　結論と今後の課題

本研究では，モデル検査法によるドローン物流システムの近似スケジューリングの手法を提案した．その具体的なアイディアは，ドローンの飛行エリアを区切る複数のマスを 1 つの新しい大きなマスへと変換することで，状態数を減らすというものである．さらに，その大きなマスの内部のスケジューリングを行うことで，スケジュールの近似解を生成することができる．この手法により，モデル検査法ではスケジュール生成が難しかった規模の物流システムでも，スケジューリングを成功させることができた．

今後の課題としては，抽象化を複数段階行う必要のある大規模な飛行エリアでも本論文の提案手法を適用したいと考えている．また近年，荷物の輸送量が増加する中で，複数のドローンが隊列を編成して 1 つの荷物を運ぶような新しいドローン物流システムが開発されている．このような隊列飛行する物流システムでは，ドローンの台数が多くなったり，ドローンの隊列の編成の要素が加わったりすることで，調べる状態数が増加することが懸念される．このように，ドローンの台数が増えたときにも，モデル検査法によるスケジューリングが可能となるような抽象化の方法の研究も行なっていく予定である．

参 考 文 献

[1] G. Behrmann, K. G. Larsen, and J. I. Rasmussen.: Optimal Scheduling Using Priced Timed Automata, *SIGMETRICS Performance Evaluation Review*, vol. 32(4),2005, pp. 34–40.

[2] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi.: Uppaal - a Tool Suite for Automatic Verification of Real-Time Systems, *International Hybrid Systems Workshop (HS 1995)*, 1995, pp. 232–243.

[3] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu.: Bounded Model Checking, *Advances in Computers*, vol.58, 2003, pp. 118–149.

[4] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta.: The nuXmv Symbolic Model Checker, *26th International Conference on Computer Aided Verification (CAV 2014)*, 2014, pp. 334–342.

[5] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri.: NuSMV: A New Symbolic Model Verifier, *11th International Conference on Computer Aided Verification* CAV 1999, 1999, pp. 495–499.

[6] E.M.ClarkeJr., O.Grumberg, D.Kroening, D.Peled, and H.Veith.: Model Checking (2nd edition), *The MIT Press*, 2018.

[7] A. Fehnker.: Scheduling a Steel Plant with Timed Automata, *6th International Conference on Real-Time Computing Systems and Applications (RTCSA' 99)*, 1999, pp. 280–286.

[8] G.J.Holzmann.: The SPIN Model Checker - Primer and Reference Manual, *Addision Wesley*, 2004.

[9] T. C. Ruys.: Optimal Scheduling using Branch and Bound with SPIN 4.0, *Interna- tional SPIN Workshop on Model Checking of Software (SPIN 2003)*, 2003, pp. 1–17.