物流情報システムにおける
業務効率化とセキュリティ強化に関する研究

提出先　　大阪大学大学院情報科学研究科

提出年月　2023 年 9 月

古家 直樹

研究業績

A. 学術論文誌論文

1. 古家直樹，永原聡士，末光一成，植木隆雄，荒宏視，嶋津泰毅 : “在庫格納期間を加味した逐次処理型の倉庫内格納先推奨技術の開発,” 日本経営工学会論文誌, Vol.70, No.2, pp.82-93 (2019)

2. 古家直樹，長谷川学，小坂忠義，薦田憲久 : “分割ルートハッシュ方式を用いたブロックチェーン利用トレーサビリティ管理システム,” 電気学会論文誌C, Vol.141, No.10, pp.1101-1114 (2021)

3. 古家直樹，末光一成，薦田憲久，藤原融 : “格納期間評価による倉庫内の在庫移動方式,” 日本経営工学会論文誌, Vol.73, No.4, pp.210-221 (2023)

B. 国際会議

1. Furuya, N., Komoda, N., and Fujiwara, T. : “Improvement of Area Priority Setting of Sequential Processing Storage Location Recommendation Technology Considering Stock Storage Period,” in *Proceedings of 26th International Conference on Production Research 2021 (ICPR2021)*, No.3, pp.1-6 (2021)

C. 学会講演

1. 古家直樹，永原聡士，末光一成，植木隆雄，荒宏視，嶋津泰毅 : “在庫格納期間を加味した逐次処理型の倉庫内格納先推奨技術の開発,” 日本経営工学会 2017 年秋季大会予稿集，pp.160-161 (2017)

2. 古家直樹，長谷川学，小坂忠義，薦田憲久 : “分割ルートハッシュ方式を用いたブロックチェーン利用トレーサビリティ管理システム,” 電気学会情報システム研究会予稿集，pp.85-90 (2020)

3. 長谷川学，古家直樹，小坂忠義 : “データ秘匿性を考慮した物流トレーサビリティデータ検索技術の開発,” 電気学会情報システム研究会予稿集，pp. 97-101 (2020)

4. 古家直樹，矢内直人，藤原融 :“関数型暗号を用いた秘匿配送マッチングの TEE による実現, ” 情報処理学会第 99 回 CSEC・第 49 回 SPT・第 98 回 EIP 合同研究発表会予稿集，2022-CSEC-99 (1), pp. 1-8 (2022)

内容梗概

本論文は，筆者が 2015 年 4 月から 2023 年 9 月まで，（株）日立製作所研究開発グループ，ならびに 2020 年 10 月から 2023 年 9 月まで大阪大学大学院情報科学研究科マルチメディア工学専攻在学中に行った，物流情報システムにおける業務効率化とセキュリティ強化に関する研究成果をまとめたものである．

物流情報システムは，物流業務を効率化する目的で用いられ，その提供機能は主に倉庫内の作業計画や在庫配置などを管理する倉庫管理，SC（Supply Chain）の商品の取り扱い履歴を管理するトレーサビリティなどの追跡管理，配送計画や荷主がトラックを直接手配できない場合などにトラックを手配する配送マッチングを管理する配送管理に分類される．また SC に関連するメーカ，倉庫，小売店等の各企業の拠点やトラックから集めたデータが用いられる．物流情報システムでは，提供機能による業務効率化の効果を高めることが重要な課題であるが，それだけでなくシステム全体のセキュリティの強化も求められる．特に追跡管理のトレーサビリティと配送管理の配送マッチングについては，複数のステークホルダが関わることから，データの耐改ざん性，秘匿性に関するセキュリティ強化が求められる．

まず，倉庫管理が対象とする倉庫業務では出庫作業の工数が大半を占めることから，倉庫の保管エリアにて出庫工数の少ない在庫配置を実現するための格納先決定方法が求められる．ここで商品は PL （パレット：Pallet）に載せられて，保管エリアの PL ラックと称す棚のロケーション（PL ラックで 1 つの PL が格納される間口のこと．以降ロケと称す．）に格納される．格納先決定とは，商品が載った PL の格納先決定を意味する．また一つの PL には，別々に入荷した複数の商品が格納に利用するロケ数を節約する目的で混載されることがある．ここで作業員が出庫のために在庫を取りに行くことを出庫タッチと称す．保管エリアでは出庫出口からの距離が近いエリアで PL ラックの低い段ほど出庫工数が少ない．ここで保管エリアを上から見た時に複数の区画に区切った各区画をエリアと称し，各ロケはどのエリアにあるかと，PL ラックの何段目にあるかで表すものとし，これをエリア段と称す．既存の格納先決定方法では，出庫タッチ数の多い PL を出庫工数の少ないエリア段である優先度高エリア段に配置する．しかし，PL

上の商品はロット順で出庫されるため，現在から同一商品の先のロットを含めて PL 上の全商品が無くなるまでの期間である格納期間を考慮しないと，PL が優先度高エリア段を無駄に占有することがある．また，優先度低エリア段に格納した PL でも，高出庫タッチ数で格納期間が短くなれば, 出庫工数削減のために在庫移動させることが望ましい．そこで，格納対象 PL 上の商品の格納期間と出庫タッチ数を，保管エリアの在庫データと過去の各商品の出庫実績をもとに混載も加味して算出し, 出庫タッチ数が多く出庫が早く進み格納期間が短い PL を優先度高エリア段に格納および在庫移動する格納先決定方式 SPLP (Storage Period based Location Planning Method) を提案する．SPLP は保管エリアへの PL の格納時の格納先決定と，格納済みの PL の在庫移動の 2 つの機能を持つ. 稼働中の倉庫のデータを用いたシミュレーションによって出庫工数削減効果を検証し, 既存手法として PL に対する出庫タッチ数のみを評価して格納先エリア段を決定し，在庫移動は行わない方法に比べて出庫工数を削減できることを示す.

次に，追跡管理のトレーサビリティと配送管理の配送マッチングでは，前者は SC の各企業，後者では不特定多数の荷主とトラックがそれぞれ関係し，共に複数のステークホルダが関わるサービスであることから，データの耐改ざん性，秘匿性に関するセキュリティ強化が求められる．また，サービス品質を担保するために複数ステークホルダのデータを難なく処理するための処理性能が求められる. まずトレーサビリティ管理では, 商品がメーカで製造され梱包箱に納められた後に，PL に載せられて倉庫に出荷され，その後店舗に出荷され販売される．またその各工程で TR.データ（トレーサビリティデータ）が生じる．ここで倉庫において梱包箱などに対して複数メーカの商品の混載が生じた場合，各メーカには他社商品の情報を秘匿する必要がある．またトレーサビリティ管理システムでは，TR.データの改ざんを防ぐ目的で BC (Blockchain) がよく用いられる．ここで経済産業省のコンビニ電子タグ 1,000 億枚宣言実現時には，全国のコンビニ店舗だけでも秒間約 3,600 トランザクションの TR.データの BC 書き込み性能が求められ，これを達成することが処理性能の目標となる．しかし既存の BC 基盤では処理性能が不足しており, 高速な TR.データの書き込みを実現する施策が必要である. そこで TR.データをメーカごとに分けてハッシュを生成し, それら複数のハッシュから生成したルートハッシュを BC の 1Tx に複数書き込む分割ルートハッシュ方式を提案する．BC を

用いた評価環境を構築して書き込み性能を評価し，年間 1,000 億個の商品を取り扱い可能であることを示す．またデータの秘匿性を評価し，提案手法の有効性を確認する．

最後に配送マッチングでは，サービス提供者がトラックのルートを 1 日の配送が始まる前に予め入手しており，荷主から受信した追加配送依頼の追加オーダをトラックに割り当てる．ここでルートと追加オーダには配送の出発地・到着地の位置情報が含まれるが，位置情報を本人以外の他者に閲覧されると取引先企業名等が分かり望ましくない．そのため，ルートと追加オーダ中の位置情報をサービス提供者も含めた他者に秘匿しながらマッチングを実施する秘匿配送マッチングの実現が求められる．ここで，国内大手マッチングサービス会社と同等のマッチング件数を達成するためには，289 件のルートが含まれる 1 回の秘匿配送マッチングを 3.6 秒以内に成立する必要がある．そこでハードウェア的に秘密計算を高速に実施する手段である TEE (Trusted Execution Environment) によって秘密計算の一手法である関数型暗号を実現する．関数型暗号では，公開鍵で暗号化した暗号文に秘密鍵を適用すると元の平文の代わりに演算結果が取得できる．TEE を用いて関数型暗号を実現した場合，暗号化されたルートと追加オーダを TEE 内で復号して配送マッチングを実施することで，ルートと追加オーダの内容を TEE の外に秘匿して高速にマッチング処理ができるが，復号に掛かるオーバヘッドなどの処理時間は未知である．そこで TEE として一般的に普及している Intel SGX を用いた評価環境を構築して処理時間を評価し，トラック 289 台のルートが含まれる 1 回のマッチングを 3.6 秒以内に処理する目標に対して，約 2.5 秒で処理可能であることを示す．

本論文は全 5 章で構成する．第 1 章の序論では，物流情報システムの概要および課題を述べる．さらにそれら課題に対する従来研究を概観し，本研究の位置付けを明らかにした上で，本研究の方針を述べる．次に，第 2 章から第 4 章では，前述した 3 点の課題に対する提案と評価を行う．最後に，第 5 章では，結論として本研究で得られた成果を要約し，今後に残された課題について述べる．

目次

第1章
序論

1.1　研究の背景

近年，物流業界においては電子商取引の普及などによって取り扱い物量が増大し，労働力不足が深刻である[1][2]. このような状況下で, サプライチェーン（SC : Supply Chain）に関連するメーカ，倉庫，小売店等の各企業の拠点やトラックからデータを取集し，それらのデータを用いて SC 全体の物流に関わる業務を効率化する機能を提供する，物流情報システムの開発・適用が求められている[3][4].

図 1-1 に物流情報システムの全体像を示す. 物流情報システムは，主に物流業務における倉庫管理，追跡管理，配送管理に関する機能から構成され，それぞれの業務を効率化する目的で用いられる[4]. 倉庫管理は物流倉庫業務における作業計画や在庫配置，追跡管理は SC における商品の取り扱い履歴を管理するトレーサビリティ，配送管理はトラックの配送計画や荷主がトラックを直接手配できない場合などに配送可能なトラックを手配する配送マッチングといった機能から構成される. また IT 基盤ではデータの収集・管理を行う. 物流情報システムでは, 提供機能による業務効率の向上のみならず, 複数ステークホルダのデータを改ざんなく保持し, 機微情報を秘匿するセキュリティ強化も求められる[3][5].

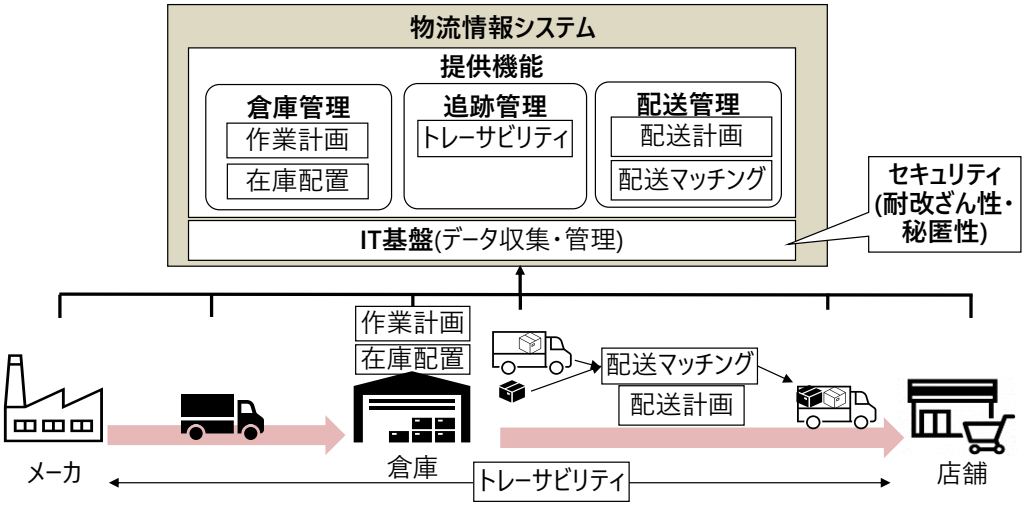

図 1-1 物流情報システムの全体像

ここで倉庫管理については，倉庫業務では出庫作業の工数が大半を占めることから，倉庫の保管エリア（倉庫に入荷した商品を格納・保管するエリア）において出庫工数の少ない在庫配置を実現するための格納先決定方法が求められる[6][7][8]．ここで商品はPL （パレット：Pallet）に載せられて，保管エリアの PL ラックと称す棚のロケーション（PL ラックで 1 つの PL が格納される間口のこと．以降ロケと称す．）に格納される．格納先決定とは，商品が載った PL の格納先決定を意味する．また一つの PL には，別々に入荷した複数の商品を格納に利用するロケ数を節約する目的で混載することがある．なお出庫工数削減方法として，出庫作業時に作業員が複数のロケを巡回する場合の巡回経路を最適化する方法もあるが，本研究では出庫作業時に巡回を行わない倉庫を対象とした格納先決定の問題を対象とする．

保管エリアでは PL ラックのロケの段数を段と呼び，下から 1 段，2 段…と数える．また保管エリアを上から見た時に複数の区画に区切った各区画をエリアと称す．各ロケはどの区画（エリア）にあるかと，PL ラックの何段目にあるかで表すものとし，これをエリア段と称す．保管エリアでは，出庫出口からの距離が近いエリアで低い段ほど出庫工数は小さく，そのようなエリア段を優先度高エリア段と称す．また作業員が出庫のために在庫を取りに行くことを出庫タッチと称す．既存手法では，出庫タッチの回数である出庫タッチ数の多い商品を優先度高エリア段のロケに格納することが多い．

しかし保管エリア内の商品は，同じ商品の出庫順序を識別するための識別子であるロットが振られロット順で出庫されることから，保管エリアに既にある先のロット順の商品が無くならないと格納した商品に出庫タッチが発生しない．そのため，各商品が現在からロット順で出庫されて無くなるまでの期間である格納期間を考慮しないと，出庫タッチ数が多い商品が載った PL でも優先度高エリア段のロケを無駄に占有してしまう．また保管エリアではロケに有る PL から全ての商品が無くならないと次の PL が格納できないため，混載 PL（複数商品の混載された PL）の格納期間は，PL 上の各商品の格納期間の最大値で評価する必要がある．さらに出庫工数削減のために，優先度低エリア段に格納された PL でも，高出庫タッチ数で格納期間が短くなれば優先度高エリア段に在庫移動させることが望ましい．以上により，混載 PL を含めた PL の格納期間を加味した格納先決定方法が課題となる．格納先決定方法は保管エリアへの PL の格納と，格納済み PL の在庫移動の両方を含む．

次に，追跡管理のトレーサビリティと配送管理の配送マッチングについては，トレーサビリティは SC の各企業，配送マッチングは不特定多数の荷主とトラックが関係し，共に複数のステークホルダが関わるサービスであることから，データの耐改ざん性，秘

匿性に関するセキュリティ強化が求められる．また単にセキュリティを強化するのみならず，サービス品質を担保するために複数ステークホルダのデータを難なく処理するための処理性能の担保が求められる．

まずトレーサビリティの課題について説明する．トレーサビリティ（追跡可能性）とは，商品の製造や原産地から販売までの履歴を記録しておき，不良品などの問題が起きた時に履歴をさかのぼって原因箇所を特定しやすくしておくこと，および商品を取り扱う作業者や一般消費者が，SC の各工程で商品が正しく品質管理されていたかを確認可能としておくことを意味する[4]．近年，食品や医薬品を中心に SC におけるトレーサビリティ管理のニーズが高まっている[9]．トレーサビリティデータ（Traceability Data．以降，TR.データと称す．）は SC の各拠点（例：メーカ，卸（倉庫），小売店）で商品の製造，入出荷などの作業が行われる度に発生する．TR.データには各作業の作業者，時刻，場所および場所の温度などの情報が記録されるが，不良品などの問題が発生し，その原因工程として特定されることを逃れる目的で，各拠点の作業者が TR.データの情報（例：温度情報など）を改ざんする恐れがある．そこで，TR.データの改ざんを防ぐ目的で Blockchain[10][11]（以降，BC と称す）が用いられることが多い[12][13].

その状況下で，TR.データの複数メーカの商品混載時の秘匿性を実現しつつ，BC への書き込み性能を担保することが課題となる．まず秘匿性について説明する．TR.データは電子タグ等が取り付けられた商品および複数商品を入れた梱包箱，および複数の梱包箱を載せた PL，かご車などの単位で発生する．梱包箱は商品が梱包される箱である．ここで，倉庫において同一の梱包箱などに複数商品が混載されると，梱包箱の取り扱い履歴を記録した TR.データに複数製造元メーカの商品の情報が記載される．あるメーカが混載の含まれた TR.データを閲覧する際に，他社商品のデータは秘匿されるべきであり，対策が必要となる．なお混載はトラック不足が叫ばれる中で[1][2]，同じ出荷先の店舗に少量の複数メーカの商品を出荷する際に，トラックの荷台のスペースを節約する必要があることから行われる．ここで書き込み性能に関しては，経済産業省は 2025 年までに国内コンビニ大手で扱う年間 1,000 億個の商品に電子タグを取り付ける「コンビニ電子タグ 1,000 億宣言」を公表している[14]．それら商品の TR.データを BC で取り扱う場合，店舗だけでも約 3,600Tx（トランザクション）/ sec の書き込み性能が求められる．既存の BC 基盤の書き込み性能は，比較的高速な Hyperledger Fabric[15]を用いた場合でも数百 Tx / sec 程度に留まり[16]，高速な TR.データの書き込みを実現する施策が必要である．以上により，トレーサビリティ管理における秘匿性と BC 書き込み性能の両立が課題となる．

最後に配送マッチングに関する課題を説明する. 配送マッチングサービスでは配送で運ぶ荷物とそれを配送可能なトラック便をマッチングする[4]．ここで荷物はトラックで運搬される品物全体を表し, 上述の在庫管理とトレーサビリティで取り扱う商品以外にも，個人間でやりとりする宅配の小包などを含む．配送マッチングサービスでは，サービス提供者がトラック便のルート情報を保持しており, 荷主から受信した追加オーダをトラックに割り当てる．ルートと追加オーダには配送の出発地・到着地の位置情報が含まれる．トラックと荷主にとって，ルートおよび追加オーダの出発地・到着地の位置情報をサービス提供者も含めた他者に閲覧されると, Web 等で検索すれば取引先企業名等が分かり望ましくない．そこで，ルートと追加オーダ中の位置情報をサービス提供者も含めた他者に秘匿しながらマッチングを実施する秘匿配送マッチングの実現が求められる．

ここで, 国内大手マッチングサービス会社と同等のマッチング件数を達成するためには，4 章で説明する通り 289 件のルートが含まれる 1 回の秘匿配送マッチングを 3.6 秒以内に成立する必要がある．秘密計算の手法の例として，暗号文のままで線形演算と乗算が可能な準同型暗号[17]があるが，同手法ではこの制約を満たすのは困難である．一方で秘密計算を高速に実現する方法として, ハードウェア的に秘密計算を高速に実施する手段である TEE (Trusted Execution Environment) [18]によって秘密計算の一手法である関数型暗号[19]を実現する方法が知られている[20]．ここで TEE とは，ユーザが安全にプログラムを実行できる保護領域であり，例として ARM の Trust Zone[21]や Intel の SGX[22]が有る．また関数型暗号は鍵生成者(Key Manager) が秘密鍵に演算を埋め込める暗号化方式であり, 公開鍵で暗号化した暗号文に秘密鍵を適用すると元の平文の代わりに演算結果が取得できる[19]．TEE による関数型暗号では，TEE の中で暗号文を平文に復号して演算を実施することで高速な処理が期待でき，また平文は TEE 外から閲覧不可となる．また，鍵生成者が認証した処理以外は実施不可のため，TEE の処理を改ざんされるなどして秘密鍵が外部に不正に流出することもない．その一方で，TEE 内でルートと追加オーダを平文に復号する処理などの処理時間のオーバヘッドが懸念される．そこで，秘匿配送マッチングに TEE による関数型暗号を適用した場合の処理時間を明らかにする必要がある．

これらを踏まえ，本論文では以下の 3 点の課題に着目する．

課題(1)　混載 PL を含めた PL の格納期間を加味した倉庫内在庫の格納先決定方法の確立

課題(2)　トレーサビリティ管理における秘匿性と BC 書き込み性能の両立

課題(3) 秘匿配送マッチングを TEE で実現した際の処理時間の評価

本研究の上記三点の研究課題について，図 1-2 に示す.

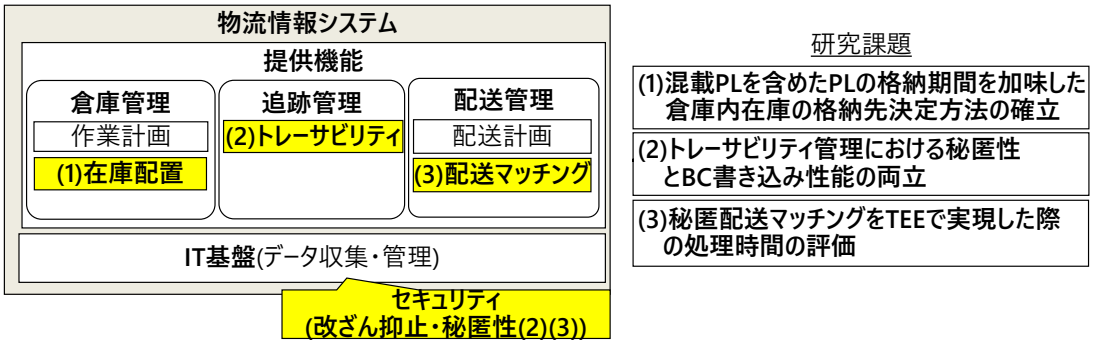

図 1-2 物流情報システムにおける研究課題

1.2 関連研究

1.2.1 倉庫内の格納先決定方法の関連研究

倉庫内の出庫工数削減を目的とした研究には，倉庫内の格納先決定方法に関する研究と，作業計画に関する研究に大別される．また格納先決定方法に関しては，保管エリアに格納前の商品を格納する時に用いる方法と，格納済の商品を在庫移動する際に用いる方法の 2 つが有る．前者の格納前商品の格納先決定方法については，まず文献[23][24]に記載の COI (Cuber per Order Index) を用いた方法がある．COI は「ある商品の倉庫内の総在庫数 / ある期間の出庫タッチ数」で定義され，COI が小さい商品ほど出庫出口近くに格納する．次に，ABC 分析[7]などを用いて出庫タッチ数などの基準によって商品を分類して格納先を決める方法である Class-based Storage がある[25][26].

別の方法として，商品間の依存関係を考慮した格納方法である Family Grouping がある．例えば，文献[27][28][29][30]では出庫タッチ数と商品間の共起頻度（商品同士が同じ出庫オーダで注文される頻度）を同時に評価して，高出庫タッチ数のグループに属する商品を出庫出口近くのロケに格納し，かつ依存関係の強い商品同士を近くに配置する．ここで商品には，出庫タッチ数は多いが一回の出庫タッチでピックする個数が少ないものも存在する．そのような商品は格納期間が長く，出庫タッチが掛かるまで優先度高エリア段を無駄に占有することになり，他の優先度高の PL を優先度高エリア段に格納できなくなる．上記の[23]-[30]の方法では，格納期間を考慮していないため，そのような問題が起こり得る．また複数商品が PL に混載された場合を考慮していない.

また，文献[31]では DOS (Duration of Stay) という指標に基づき格納先を決める．DOS は「商品のロット順 i / Demand Rate（商品の一定期間の出庫数)」で定義され，DOS が小さい商品ほど早く出庫されるため優先度高エリア段に格納する．例えば，ロット順が 1 番目で一定期間（例として 2 週間）の出庫個数が 200 個の商品の Demand Rate は 1 / 200 となる．しかし，DOS では複数商品の混載は考慮されていない．また DOS では各商品の格納時の個数は全ロット順で同じと仮定しているが, 実際には格納時の個数がロット順によって異なることが多い. そのため, 各ロット順の個数を考慮する必要がある．

次に，在庫移動に関する従来研究を述べる．Jaikumar らは，Class-Based Storage によって出庫タッチ数が多い商品を優先度の高いロケに移動させている[32]．Kofler らは出庫タッチ数と共起頻度の双方を加味して在庫を移動させており, 高出庫タッチ数の商品を出口近くに移動させ, かつ同じ出庫オーダで出庫される商品同士を近くに配置する[33][34]．また Chen らは自動倉庫の在庫の格納・取り出し時間を削減するために在庫移動を行っており[35]，組み合わせ最適化問題において，局所最適解に陥ることを防止する Tabu Search アルゴリズム[36]を用いて在庫移動に掛かる時間を最小化している．次に Chen らは[37]にて問題を複数に分割し近似の最適値を定義する近似動的計画法[38]を用いて，自動倉庫において出庫が掛かった PL の配置を最適化している．しかし[32]-[35][37]のいずれも，格納期間および異なる商品の混載が考慮されていない．

最後に，出庫工数削減を目的とした作業計画に関する研究を述べる．作業計画の研究では，作業員が出庫時に巡回でピックする場合に，ピック指示を行う出庫オーダ（1 回の巡回でピックする商品が記載されたオーダ）に割り当てる商品を最適化し，さらに 1 回の出庫オーダ中のピック順序を最適化することで巡回経路を最短にする方法がある[39]．また文献[40]では，出庫タッチ数が多い商品と少ない商品とで出庫オーダを分ける方法が取られている．

1.2.2 トレーサビリティ管理の秘匿性と書き込み性能の関連研究

物品のトレーサビリティ管理の研究は BC の登場前から行われて来た．例えば Staake らは TR.データの国際標準である EPC (Electronic Product Code) データ[41][42]を対象に，TR.データを企業ごとの秘密鍵で暗号化することで他者からの耐改ざん性と秘匿性を保証している[43]．しかし [43]の研究では複数メーカの商品が混載する場合の秘匿性には

言及していない．なお EPC では TR.データのフォーマットが規定されているが，セキュリティ（耐改ざん性，秘匿性）の担保方法については規定が無い．

次に BC を用いた研究事例を示す．Westerkamp らは，パブリック型 BC 基盤の Ethereum[44]を用いた事例を示している[45]．Toyoda らは，物品に RFID (Radio Frequency Identification) タグを付け，トレーサビリティを BC で管理することで，偽物へのすり替えを防ぐ方法を述べている[46]．Lin らも EPC を用いた食品トレーサビリティの研究を行っている[47]．しかし，これらは複数メーカの商品が混載される場合の TR.データの秘匿性について言及していない．

また Shakhbulatov らは，SC における二酸化炭素排出量を BC で追跡する研究にて，予め閲覧者をグループ分けしておきデータのアクセス範囲を限定することでデータの秘匿性を実現している[48]．しかしこの方法では，本研究で想定する全国のコンビニ商品のように不特定多数の企業が参画し，出荷先も事前に分からない状況に対応困難である．また，IBM は BC を活用して食品のトレーサビリティを対象とした Food Trust のサービスを 2018 年に開始している[49]．BC 基盤として Hyperledger Fabric を採用し EPC データを扱うが，書き込み性能や複数メーカの商品が SC の過程で混載された場合のデータの秘匿性については公表されていない．

なお，SC 以外の分野で機微なデータの秘匿を実現した例として，クレジットカードのオンライン決済の通信プロトコルである SET[50]がある．SET では，カード所有者がインターネット上の店舗に送る発注情報と，カード会社に送る決済関連情報を分離することで，カード所有者の発注内容をカード会社に秘匿しながら決済ができる．

1.2.3 TEE を用いた秘匿配送マッチングの関連研究

秘密計算の一般的な手法として，暗号文で線形演算と乗算が実施可能な準同型暗号（完全準同型暗号）[17]が知られている．準同型暗号では，暗号文のまま処理を行う性質上，処理時間のオーバヘッドが大きいことが課題である[51]．

処理時間の課題に対して，TEE を用いることでハードウェア的に秘密計算を行うことで解決を図る研究が多数報告されている．例えば Fisch らは，TEE として SGX を用いて関数型暗号を実現する IRON を提案しており，TEE を用いることで従来の関数型暗号よりも数桁高速な処理が期待できると述べている[20]．また IRON を拡張した研究事例として，一定時間が経過するまで復号できないことを保証した関数型暗号である関数型タイムリリース暗号[53]の提案や，システムの内部状態に依存し，かつランダム化され

た関数型暗号を実現した Steel の提案がある[54]．[53][54]とも IRON を拡張して関数型暗号で扱うことのできる関数の種別を拡大することを目的としているが，本研究の配送マッチングのように具体的な適用先の問題（アプリケーション）に対して適用評価を行ったものではなく，[53][54]で開発した暗号が実用的であるかの評価は未実施である．

次に TEE として SGX を用いた研究事例で，関数型暗号以外のものを示す．まず，個人のゲノムデータを，サーバ管理者に秘匿した状態で解析を行う事例が知られている[55][56]．これらでは，SGX の TEE にメモリサイズの制限があることから，分割したゲノム情報の暗号文を高速かつ安全に TEE 内に展開するために，データの牽引化とパス型 ORAM[57]（Oblivious RAM：サーバからアクセスパタンを秘匿しつつランダムアクセスを行うための暗号学的技術）を用いている．また Felsen らは，SGX を用いて 2 つのパーティがお互いの入力値は明かさずに，公開された関数の出力値を得る Secure Function Evaluation[58]，および片方のパーティが関数を明らかにせずに，もう片方のパーティからの秘匿化された入力値に対する結果を得る Private Function Evaluation[59]を実現しており[60]，処理が複雑になっても Yao のガーブル回路[61]と GMW[62]に比べて高速であることを述べている．[55][56][60]とも，入力データを暗号化して TEE に送り平文で処理を行う点は本研究と共通である．一方で，関数型暗号で考慮される鍵生成者が不在なため，仮に TEE 内の処理をサービス提供者（サーバ管理者）が書き換えた場合，TEE 内の復号鍵を入手できることを考慮していない課題が有る．

1.3 研究の方針

前節までに述べた課題を踏まえ，本研究の方針を説明する．

(1) 格納期間を加味した在庫格納先決定方式の提案

課題(1) に対して，格納 PL（格納対象商品が載った PL）および移動対象 PL（保管エリアに格納済みの PL で，在庫移動を行う対象の PL）上の各商品の過去の出庫実績を WMS（Warehouse Management System：倉庫管理システム）のデータをもとに評価し，出庫タッチ数が多く格納期間が短いと予測される PL を優先度高エリア段に格納，在庫移動する在庫格納先決定方式 SPLP (Storage Period based Location Planning Method) を開発する．SPLP では保管エリアへの PL の格納と在庫移動の双方を対象に，PL 上の商品の混載も考慮して，出庫タッチ数と格納期間から格納 PL の評価値を算出し，保管エリアに格納済の PL である在庫 PL の評価値との順位比較により格納先のエリア段（以後，

格納先エリア段と称す）を決定する．シミュレーションによって，既存手法として PL に対する出庫タッチ数のみを考慮して格納先を決定し，在庫移動は行わない方法と比較した出庫工数削減効果を検証する．

(2) 分割ルートハッシュ (Root Hash：RH) 方式による BC 利用トレーサビリティ管理システムの提案

課題(2) に対して，TR.データのデータ秘匿性と BC の書き込み性能を両立する書き込み方式である，分割 RH 方式を用いたトレーサビリティ管理システムを提案する．分割 RH 方式では，複数商品の混載が生じている TR.データを，商品の製造元メーカごとに分けた後にハッシュを生成し，それら複数のハッシュから生成したルートハッシュを BC の 1Tx に複数書き込むことで，データの秘匿性と BC への書き込み性能を両立する．

TR.データの秘匿性を既存手法との比較により考察するとともに，高速な BC 基盤である Hyperledger Fabric を用いて，クラウド環境上で書き込み性能の評価を行い，コンビニ電子タグ 1,000 億宣言で想定される書き込み性能を達成できることを検証する．

(3) TEE を用いた関数型暗号による秘匿配送マッチングシステムの処理時間の評価

課題(3) に対して，本研究では IRON[20]を拡張して，配送マッチング問題に適用し TEE として SGX を用いて関数型暗号による秘匿配送マッチングシステムの評価環境を構築して処理時間を評価する．それによって，秘匿配送マッチングシステムが，国内大手マッチングサービス会社と同等のマッチング件数を達成するために必要となる，トラックのルートが 289 件存在するマッチングを約 3.6 秒以内という制約を満たせることを検証する．また，サービス提供者を含む他者にルートと追加オーダが秘匿できることを考察する．

1.4 本論文の構成

本論文では，第 2 章以降を以下のとおり構成する．

第 2 章では，文献[63]-[66]に基づき，格納期間を加味した在庫格納先決定方式を提案し，シミュレーションにより出庫工数削減効果を評価する．

第 3 章では，文献[67]-[69]に基づき，分割 RH 方式による BC 利用トレーサビリティ管理システムを提案し，クラウド環境上で TR.データの書き込み性能の評価を行うとともに，提案手法のデータの秘匿性を考察する．

第 4 章では，文献[70]に基づき，TEE を用いた関数型暗号による秘匿マッチングシステムの評価環境を用いた処理時間の評価を行うとともに，秘匿性について考察する．

第 5 章では，結論として本研究で得られた成果を要約し，今後の課題を述べる．

第2章　格納期間を加味した在庫格納先決定方式

2.1　緒言

物流業界においては人手不足が深刻であり，倉庫業務においても雇用の確保が難しくなる中で作業員の業務効率化が求められている．倉庫内では出庫工数の比率が大きく，倉庫内の作業工数の半分以上を占めることから出庫工数削減が重要である．そこで，出庫工数が少なくなる在庫配置を実現するための倉庫内在庫の格納先決定技術が求められる．

ここで商品はPLに載せられて，保管エリアのPLラックと称す棚のロケに格納される．格納先決定とは，商品が載ったPLの格納先決定を意味する．また一つのPLには，別々に入荷した複数の商品を格納に利用するロケ数を節約する目的で混載することがある．出庫工数が少なくなる格納先決定方法には様々あるが，出庫のために商品を取りに行く回数を意味する出庫タッチ数の多い商品が載ったPLを出庫工数が少ない優先度高エリア段に格納することが一般的である．しかし倉庫の在庫はロット順で出庫されることから，保管エリアに既にある同一商品の先のロット順の商品が無くなるまで出庫タッチが発生しない．そのため，現在から商品がロット順で出庫されて無くなるまでの期間である格納期間を考慮しないと，出庫タッチ数が多い商品が載ったPLでも優先度高エリア段のロケを無駄に占有することが起こり得る．また優先度低エリア段に格納されたPLでも，優先度高エリア段に移動させることで出庫工数を減らすことができる．

そこで商品が載せられたPLに対して，格納期間を加味して格納先のロケの指示を行う格納先決定方式SPLPを提案する．本方式では，過去の商品ごとの出庫実績と保管エリアの在庫データから，混載も加味して格納PLの出庫タッチ数と格納期間を算出し，高出庫タッチ数で格納期間の短いPLに優先度高エリア段を割り当てる．また優先度低エリア段に格納されたPLも高出庫タッチ数で格納期間が短くなれば，同様に優先度高エリア段に在庫移動を行う．これにより，出庫工数の削減を実現する．

本章で対象とする倉庫は，保管エリアからの出庫のパタン（以後，出庫パタンと称す）として 2.2.1 節で説明する CS (Case) 出庫と補充出庫の 2 パタンがあるものとする．また 1 つのロケからのピッキングで保管エリアの作業で用いられるフォークリフトに載せられる量の限度になり，作業員は出庫の際に商品を搬出する出庫出口に都度出庫対象

商品を運搬するものとする．本章ではシミュレーションによって出庫工数削減効果を検証し，既存手法として PL に対する出庫タッチ数のみを評価して格納先を決定する方法に比べて，提案手法が出庫工数を削減できることを確認し，提案手法の有効性を示す．

以降，2.2節で倉庫内の格納先決定の課題，2.3節では提案の格納先決定方式SPLP，2.4節ではSPLPの処理フロー，2.5節ではシミュレーションによる出庫工数削減効果の評価，2.6節で考察，2.7節でまとめを述べる．

2.2　倉庫内の格納先決定の課題

2.2.1　対象とする倉庫

本章で対象とする倉庫の保管エリアについて，保管エリア内の作業の様子を示した図 2-1 を用いて説明する．保管エリアでは作業員がフォークリフトを用いて格納・出庫作業を行う．保管エリアで保管される商品は CS（ケース：Case）と呼ばれる梱包箱に詰められており，PL に載せられて運搬される．また図 2-1 の両端に並んでいる商品の格納先の棚が PL ラックである．PL ラックは 4 段構成となっており，低い段から 1 段，2 段と数える．作業員は PL ラックに商品を PL に載せられた状態で格納する．PL ラックで 1 つの PL が格納される間口をロケと称する．各ロケのサイズ（縦・横・高さ）および各 PL のサイズは同じであり，各ロケに 1PL ずつ格納される．ロケ数は約 3,000（3,085）ロケで，概ねその約 3/4 の 2,300PL 以上が格納されている．各 PL に各商品の CS を何個まで載せられるかは，各商

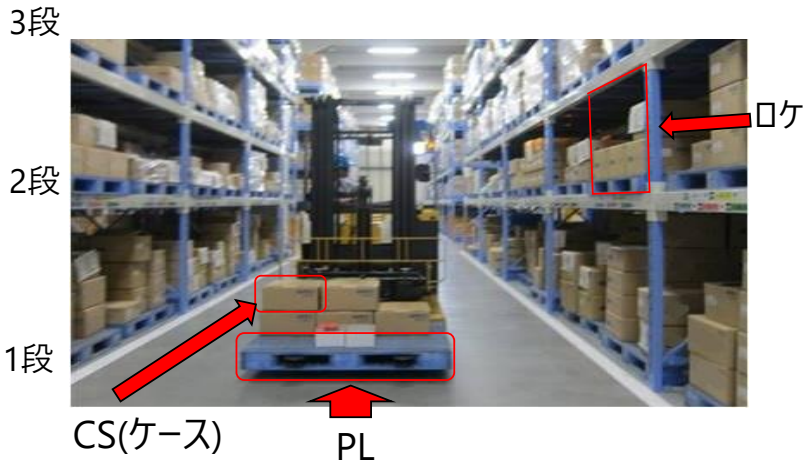

図 2-1 保管エリア内の作業の様子

品の 1CS 当たりの容積によって異なるが, 作業員の判断でロケのサイズに納まる個数を PL に載せて格納する.

次に, 倉庫を上から見たレイアウトである図 2-2 を用いて, 倉庫作業の流れを説明する. 同図で保管エリアのレイアウトは簡略化している. 本章で扱う倉庫は, 商品を一定期間保管した後に出庫する DC (Distribution Center) 型と呼ばれる倉庫である. 倉庫で保管される商品は, 入荷バースで検品された後, PL に載せられて保管エリアに格納される. PL には作業員の判断で複数商品が混載されるため, PL に商品が載ってからでないと格納先ロケを決定できない. なお混載は, 格納に利用するロケ数を節約するために国内では一般的に行われている.

ここで, 格納されている商品を当該倉庫から他の施設へ搬出することを出荷という. 一方, 倉庫の保管エリアからその外に商品を搬出することを出庫という. 出荷には, 大量出荷の場合に商品を CS のまま出荷する CS 出荷, および少量出荷の場合に商品を CS から開梱して出荷するピース出荷の 2 つがある. CS 出荷の場合, 商品の入った CS が保管エリアでピッキングされた後, 出荷バースから出荷される. 出荷バースは出荷対象商品がトラックに載せられる作業場である. 一方ピース出荷の場合, 商品はピース出荷に備えて CS から開梱した状態でピッキングエリアにて保持され, ピッキングエリアからピッキング後に出荷バースから出荷される. ピッキングエリアの在庫が少なくなった場合には, 保管エリアからピッキングエリアへの補充が行われる.

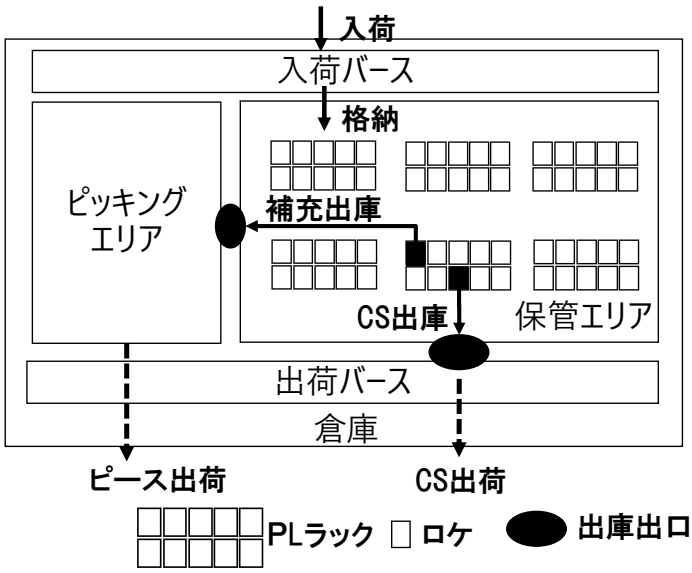

図 2-2 対象とする倉庫を上から見た図

ここで, CS 出荷における保管エリアから出荷バースへの移動と保管エリアからピッキングエリアへの補充の作業における商品の移動を出庫と呼ぶ．前者を CS 出庫，後者を補充出庫と呼ぶ．なおピッキングエリアにおいては，保管エリアに比べると各商品は少量ずつしか置かれないので，商品間の格納期間のばらつきが小さく，在庫配置を工夫しても効果が小さい．そのため本章では，保管エリアの在庫配置を研究対象とする．また前述のように，保管エリアには CS 出庫と補充出庫の 2 つの出庫のパタン（以降，出庫パタンと称す）があることから，本章では出庫工数として CS 出庫と補充出庫の双方の工数を削減することを目的とする．

2.2.2 出庫作業の流れと出庫工数

出庫作業の流れを説明する．作業の流れは CS 出庫，補充出庫とも同一である．本章では，出庫の際作業員が都度出庫出口に出庫対象商品を置きに行く倉庫を対象とする．まず作業員は，出庫出口からフォークリフトで出庫対象商品の有るロケの前に向かう．フォークリフトには空の PL が載っており，ピック対象商品が 1 段目に有れば，作業員が PL ラックから手作業で商品をフォークリフト上の PL に載せる．一方 2〜4 段目の場合は，フォークリフトを用いて PL ラック上の PL を降ろした後，作業員が手作業で商品をフォークリフト上の PL に載せる．その後フォークリフトで元の商品の載った PL を PL ラックのロケに戻す．ここでロケに有った PL の全商品が無くなった場合でも，PL が通路を妨げないように元のロケに戻され，出庫作業完了後にその PL は回収される．このように，作業員が出庫対象商品の有るロケの前に到着した後に，

・フォークリフトで PL を降ろす（2 段目以上の場合）
・商品を PL から取得してフォークリフト上の PL に載せる
・商品の載っていた PL を元のロケに戻す（2 段目以上の場合）

という一連の工程を在庫取得と称す.

その後作業員はフォークリフトを運転して出庫出口に戻り，PL 上の商品を出庫出口に置く．出庫出口は図 2-2 に示すように，CS 出庫と補充出庫とで場所が離れている．本章では,

・作業員がフォークリフトで，出庫出口からピック対象商品の有るロケの前に移動する
・ピック対象商品の有るロケから出庫出口に，商品をフォークリフトで運ぶ

という一連の工程を運搬と称す．すると，出庫作業は在庫取得と運搬の 2 つの工程から構成され，1 回あたりの出庫工数は,

出庫工数（人分） ＝ 在庫取得工数（人分）+ 運搬工数（人分）（２－１）
と表すことができる.

ここで 1 回の在庫取得に掛かる時間は 1 段目が最も小さく，2 段目以上においてはフォークリフトの腕を伸ばす時間が生じることから，段数が高いほど時間が掛かる．また運搬についても出庫出口からのピック対象商品の有るロケまでの距離（通路を加味した動線距離）に比例する．そのため，保管エリアでは，出庫出口から近く低い段のロケほど出庫に掛かる時間は少ない．ここで出庫作業は 1 回の出庫（在庫取得と運搬）につき 1 人が行うため，1 回の出庫に掛かる時間を t 分とすると，出庫 1 回あたりの出庫工数は 1 * t = t（人分）となる．そのため 1 回あたりの出庫工数は，1 回あたりの出庫時間と等価とみなすことができ，出庫出口から近く低い段のロケほど出庫工数は少ない.

表 2-1 に，対象倉庫で計測した，段ごとの在庫取得の所要時間を工数に換算した 1 回あたりの在庫取得工数を示す．1 段目ではフォークリフトを用いないことから，最も在庫取得工数が小さく，2 段目以降は段数が増えるほど在庫取得工数が増える．これは高い段ほどフォークリフトのリフト（腕）を伸ばす時間が掛かるためである．但し，腕を高く伸ばすほど腕の速度が増すため，3 段目と 4 段目の在庫取得工数の差は小さい.

2.2.3 格納先決定に関する課題

本章では, 保管エリアを上から見た時に複数の区画に区切った各区画をエリアと称す. また各ロケはどの区画（エリア）にあるかと，PL ラックの何段目にあるかで表すものとし，これをエリア段と称す．例えば，保管エリアの中を出庫出口に近いエリアを A エリア，遠いエリアを B エリアとすると，A エリアの PL ラック 1 段目にあるロケを A_1 段のロケといった形で表記する.

従来出庫工数を削減するためには, 出庫工数の少ない優先度高エリア段に属する出庫出口から近く低い段に属するロケに, 高出庫タッチ数の商品を優先的に格納すれば良いと考えられてきた．例えば既存の格納先決定手法としては，Class-Based Storage[25][26]

表 2-1 対象倉庫における 1 回あたりの在庫取得工数

段	在庫取得工数（人分）
1	1.9
2	2.3
3	2.9
4	3.0

に基づいて予め保管エリア内のエリア段を出庫工数の低い順に優先度付けをしておき，格納 PL の出庫タッチ数を過去の出庫実績データから算出して，出庫タッチ数の多い PL から順番に優先度高エリア段に格納する方法がある．しかし，保管エリアの格納作業の実態を踏まえると，次の 2 点が課題である．

- 課題(1)　高出庫タッチ数の商品であっても，ロット順の早い同一商品が出庫されないと出庫タッチが発生しない．
- 課題(2)　出庫工数削減のために，優先度低エリア段に格納した PL でも高出庫タッチ数で格納期間が短くなれば在庫移動させることが望ましい．

課題(1) について図 2-3を用いて説明する．倉庫では，各商品に対して倉庫への入荷時に同じ商品の出庫順序を識別するための識別子であるロットが振られ，ロット順に出庫がなされる．なおロットについては，商品ごとに1回の入荷で同一番号が振られる．ここで，高出庫タッチ数の商品の載ったPLを優先度高エリア段に格納しても，ロット順の早い（先のロット順の）同一商品が無くなるまで出庫タッチが発生せず，優先度高エリア段のロケを無駄に占有してしまう．ここで図 2-3に示すように，現在からPL上の商品が出庫し終わるまでの期間を格納期間と称す．また保管エリアのロケには，PL上の全商品が無くならないと次のPLが格納できないが，混載PLの場合PL上の各商品の格納期間の最大値でPLの格納期間を評価する必要がある．

課題(2) について説明する．出庫タッチ数が多い商品の載った格納 PL でも，優先度高エリア段に空きロケがなければ優先度低エリア段に格納される．そのような格納 PL に対しては，優先度低エリア段からの出庫タッチが発生して出庫工数が増大してしまうため，優先度高エリア段に在庫移動させることが望ましい．

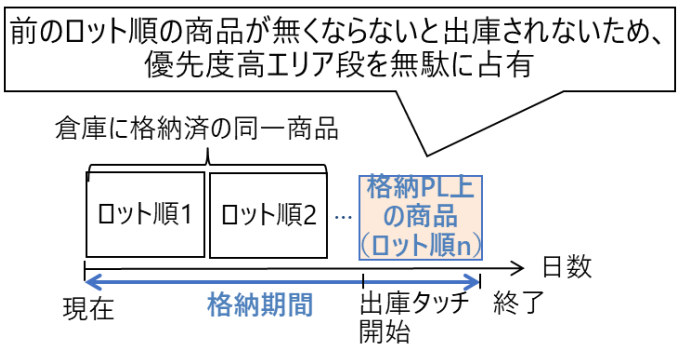

図 2-3 格納期間に関する課題

2.3 格納先決定方式 SPLP

2.3.1 アプローチ

2.2.3 節で述べた課題(1)(2) に対して，PL 上の各商品の出庫タッチ数と格納期間 (Storage Period) を WMS の過去の出庫実績および在庫データから混載も加味して算出して，出庫タッチ数が多く格納期間の短い PL を優先度高エリア段に格納する格納先決定方式 SPLP (Storage Period based Location Planning Method) を開発する．同方式は保管エリアへの PL の格納時と，格納済みの在庫 PL の在庫移動時の双方に適用する．また運搬工数は出庫対象商品の有るロケから出庫出口までの距離に依存するため，PL 上の商品は CS 出庫と補充出庫の出庫実績データを参照して，それぞれの出庫タッチ数の比率を算出して，その比率に応じていずれかの出庫出口近くのロケに格納する．ここで在庫移動については，優先度が最も高い優先度 1 のエリア段を移動先とし，保管エリア中の優先度 2 以下のエリア段に有る在庫 PL の中で，出庫タッチ数が多く格納期間の短い PL を優先度 1 のエリア段に在庫移動させる．

2.3.2 SPLP の処理概要

SPLP は，2 段階の処理から構成されるが，処理 1 は．格納時（処理 1-a）と在庫移動時（処理 1-b）とで処理内容が若干異なる．なお SPLP を用いる際は，予め保管エリア内のエリア段を出庫工数の低い順に優先度付けをしておく．

- 処理 1-a（格納時）: 格納 PL の出庫タッチ数と格納期間から格納 PL の評価値を求め，保管エリアに格納済の在庫 PL の評価値と比較した際の格納 PL の順位から格納先エリア段を決定.
- 処理 1-b（在庫移動時）: 保管エリアの優先度 2 以下のエリア段にある在庫 PL を処理 1-a と同じ評価値で日々評価し，評価値の順位が優先度 1 エリア段に設定した順位の閾値よりも高い在庫 PL を移動対象 PL として決定.
- 処理 2 : 格納先エリア段内で，CS 出庫と補充出庫の双方の出庫出口までの運搬工数の和が最小となるロケを格納先ロケとして決定.

既存の格納先決定方法では 2.2.3 節で述べたように，格納先エリア段の決定時に出庫タッチ数のみで格納 PL を評価するが，SPLP では処理 1-a, 1-b において格納期間を加味することで，出庫タッチ数は多くとも出庫タッチが当面掛からない PL が優先度高エリア段に格納されるのを防ぐ．また処理 2 によって，出庫（CS 出庫と補充出庫）実績デ

ータから，CS 出庫と補充出庫の出庫出口から出庫した回数（CS 出庫タッチ数と補充出庫タッチ数）の比率を求め，それをもとに格納先エリア段における各ロケの中で各出庫出口までの運搬工数の和が最小となるロケを格納先ロケと定める．

ここで SPLP の各処理の実行順序を説明する．格納と在庫移動のどちらを先にやるかは倉庫の運用に合わせれば良いが，対象倉庫では商品が PL に載ってからでないと格納先ロケが決定できないのに対して，在庫移動は在庫データが有れば移動対象 PL と移動先のロケを決定できる. そのため, 本章では格納よりも先に在庫移動を行う. すなわち,「在庫移動（処理 1-b→処理 2）→ 格納（格納処理 1-a→処理 2)」という順番で SPLP の処理が実行される．なお，対象倉庫のように，PL にどの商品が載るか格納の直前まで分からない倉庫は多く存在する．

2.3.3 格納先決定方法

2.3.3.1 格納先エリア段および移動対象 PL の決定方法

図 2-4 を用いて，処理 1-a における格納先エリア段の決定方法を説明する．まず，予め保管エリア内のエリア段を出庫工数の低い順に優先度付けしておく. また優先度高のエリア段から順に，格納して良い PL の評価値の順位の下限（閾値）である格納可能最下位順位を決めておく．次に格納 PL および保管エリアの全在庫 PL の評価値を算出しておき，続いて格納 PL と全在庫 PL との評価値の比較を行い，格納 PL の評価値の全在庫 PL の評価値に対する順位を算出する．その順位と，各エリア段に設定した格納可能最下位順位との比較で格納先エリア段を決定する．ここで閾値を評価値でなく，評価値の順位とするのは，評価値で閾値を設定しても各在庫 PL と格納 PL の評価値は日々変化することから，閾値で設定した値の意味が日々変わってしまい，適切な閾値とならないためである．

次に, 処理 1-a, 1-b で共通に用いる評価値の算出方法を説明する. 評価値の算出には, WMS の出庫実績データと在庫データを用いる．まず，ある PL（PL_k）上の i 番目の商品の格納期間 $P_{k,i}$ と，総出庫タッチ数 $T_{k,i,j}$ の算出方法を説明する．PL_k には，n 種類の商品が載っているものとする．

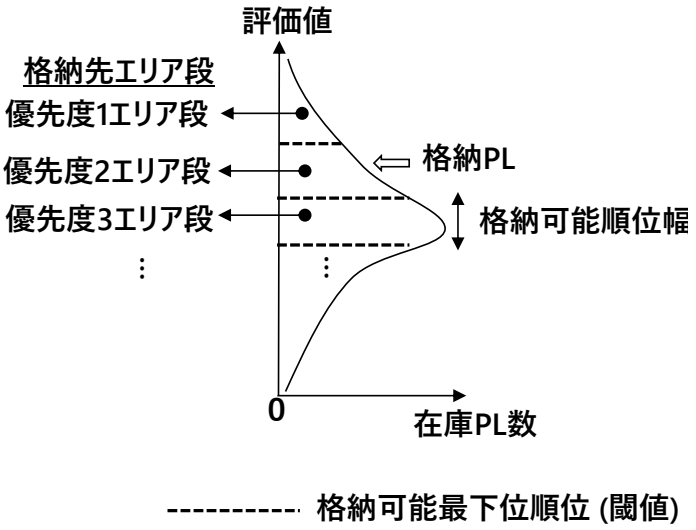

図 2-4 格納先エリア段の決定方法

PL$_k$ 上の商品 i の格納期間 $P_{k,i}$ は下式（２－２）より算出される．

$$P_{k,i} = \frac{N_{i,k}}{\bar{N}_{i,out}} \qquad （２－２）$$

・ $P_{k,i}$：PL$_k$ 上の i 番目の商品の格納期間（日）

・ $N_{i,k}$：商品 i のロット順 1 番目から，その PL$_k$ 上のロット順を含む総在庫個数（個）

・ $\bar{N}_{i,out}$：出庫実績から算出した商品 i の一日あたりの平均出庫個数（個 / 日）

次に $N_{i,k}$ は，商品 i のロット順が u 番目（$1 \leqq u \leqq l_{i,k}$）のロット順の在庫の個数を $q_{i,u}$ とすると，次式（２－３）で表される．

$$N_{i,k} = \sum_{u=1}^{l_{i,k}} q_{i,u} \qquad （２－３）$$

・ $q_{i,u}$：商品 i のロット順が u 番目の商品の個数

・ $l_{i,k}$：商品 i の PL$_k$ 上の商品のロット順

続いて，PL$_k$ 上の商品 i の出庫パタン j（j=1：CS 出庫，j=2：補充出庫）における出庫タッチ数 $T_{k,i,j}$ は，下式（２－４）にて算出される．

$$T_{k,i,j} = \frac{n_{i,k}}{\bar{n}_{i,j,out}} \qquad （２－４）$$

・ $T_{k,i,j}$：PL$_k$ 上の i 番目の商品の出庫パタン j における出庫タッチ数（回）

・ j：出庫パタン（j = 1：CS 出庫，j = 2：補充出庫）

・ $n_{i,k}$：商品 i の PL$_k$ 上の個数（個）

・ $\bar{n}_{i,j,out}$：商品 i の出庫パタン j における 1 出庫タッチ当たりの平均出庫個数（個 / タッチ）

ここで n 種類の商品が載っている PL_k の評価値 E_k は次式（２－５）で表される．

$$\boldsymbol{E_k = \frac{\sum_{i=1}^{n}\sum_{j=1}^{2} T_{k,i,j}}{max\,(P_{k,i})}} \qquad （２－５）$$

評価値 E_k は PL_k の格納期間が短く，総出庫タッチ数が多いほど高くなる．混載 PL の場合は，出庫タッチ数 $T_{k,i,j}$ は PL_k 上の全商品の総和を取り，格納期間は PL_k 上の商品の最大値を用いる．

次に，処理 1-a における格納先エリア段の決定方法を説明する．まず，保管エリアの全在庫 PL に対しても（２－５）式を用いて評価値を算出しておく．続いて格納 PL と全在庫 PL との評価値を比較し，格納 PL の評価値を全在庫 PL の評価値と比較した場合の順位を算出する．その順位と，各エリア段に設定した格納可能最下位順位との比較で格納先エリア段を決定する．なお，別の方法としては在庫 PL と格納 PL の評価値を計算しながら heap tree を構築し，各格納 PL の在庫 PL と比較した順位を求めても良い．

ここで格納先エリア段に空きロケが存在しない場合は，次の優先度のエリア段で順番に格納先ロケを探す．最下位のエリア段にも空きロケがない場合は，最初に決定した格納先エリア段よりも優先度が一つ高いエリア段から順番に探索する．そのように保管エリアの全エリア段を検索しても，空きロケが見つからず格納ができなかった PL は格納不可 PL とみなし，格納先の指示は行わない．なお実際の倉庫の運用では，格納不可 PL は保管エリアの通路等に置かれるか，格納不可が発生することが分かっている場合には，予め保管エリアの複数のロケに残っている商品同士を組み合わせて 1 つのロケに保管したり，保管エリアの在庫 PL を近隣の倉庫に転送したりすることで保管エリアに空きロケを確保することが一般的である．

また処理 1-b の在庫移動の場合には，格納時と同様に保管エリアの在庫 PL を評価値の高い順に並べ，優先度 2 以下のエリア段に有る PL の中で，評価値の順位が優先度 1 エリア段の格納可能最下位順位以上の PL を移動対象 PL として決定する．

2.3.3.2 格納先ロケの決定方法

次に処理 2 の格納先エリア段内の格納先ロケの決定方法を説明する．格納先ロケは格納先エリア段内で探索し，PL 上の各商品の CS 出庫と補充出庫の出庫タッチ数の比率から格納先ロケを決定する．すなわち，ある格納先エリア段の空きロケ p に PL_k を格納した場合に，出庫出口とロケ p との間の総運搬距離 $D_{p,k}$ を下式（２－６）で評価して，$D_{p,k}$ が最小となる空きロケを格納先ロケとする．

$$\boldsymbol{D_{p,k} = \sum_{j=1}^{2}(d_{p,j} * \sum_{i=1}^{n} T_{k,i,j})} \qquad （２－６）$$

・ $d_{p,j}$：ロケ p から CS 出庫と補充出庫の出庫出口までの通路を加味した距離

・ j：出庫パタン（$j=1$：CS 出庫，$j=2$：補充出庫）

ここで処理を処理 1-a, 1-b と処理 2 のように 2 段階に分けたのは，仮に 1 段階とする場合，格納期間に加えて，CS 出庫と補充出庫の出庫タッチ数の比率も加味した評価値が必要となり，複雑な評価値の計算となることが予想されるためである．SPLP では，まず処理 1-a, 1-b において CS 出庫と補充出庫に依らずに格納期間と CS 出庫と補充出庫の出庫タッチ数の和で PL を評価して格納先エリア段もしくは移動対象 PL を決め，その後処理 2 で格納先エリア段（在庫移動の場合は優先度 1 エリア段）内の格納先ロケを，CS 出庫と補充出庫の出庫タッチ数の比率を加味して決定することで格納先決定ロジックを単純化している．

2.4 処理フロー

2.4.1 格納時の処理フロー

図 2-5 を用いて，格納時の SPLP の処理フローを説明する．ステップ S1，S2 は前処理であり，ステップ S3～S13 が個々の格納 PL に対して行う格納先決定処理である．

S1 では，保管エリア内のエリア段を優先度付けし，各エリア段に格納可能な評価値の順位の閾値（格納可能最下位順位）を設定しておく．エリア段の分け方はロジック上任意であるが, 例えばエリアは保管エリアを双方の出庫出口から距離的に遠いエリアと近いエリアの 2 つに分け，さらに段によってエリア段を分けることが考えられる．適切なエリア段の分け方については今後の課題である．

次に，各エリア段の格納可能最下位順位の設定方法を説明する．基本的に各エリア段の格納可能順位幅（そのエリア段より一つ上の優先度のエリア段の格納可能最下位順位とそのエリア段の格納可能最下位順位との差）を広げるほど，格納 PL がそのエリア段に格納される可能性が高くなり，格納可能順位幅を広げ過ぎると，本来格納されるべきではない格納 PL が格納されてしまう．一方で格納可能順位幅を狭め過ぎると，本来そのエリア段に格納されるべき格納 PL が格納されなくなる．そこで，各エリア段の格納

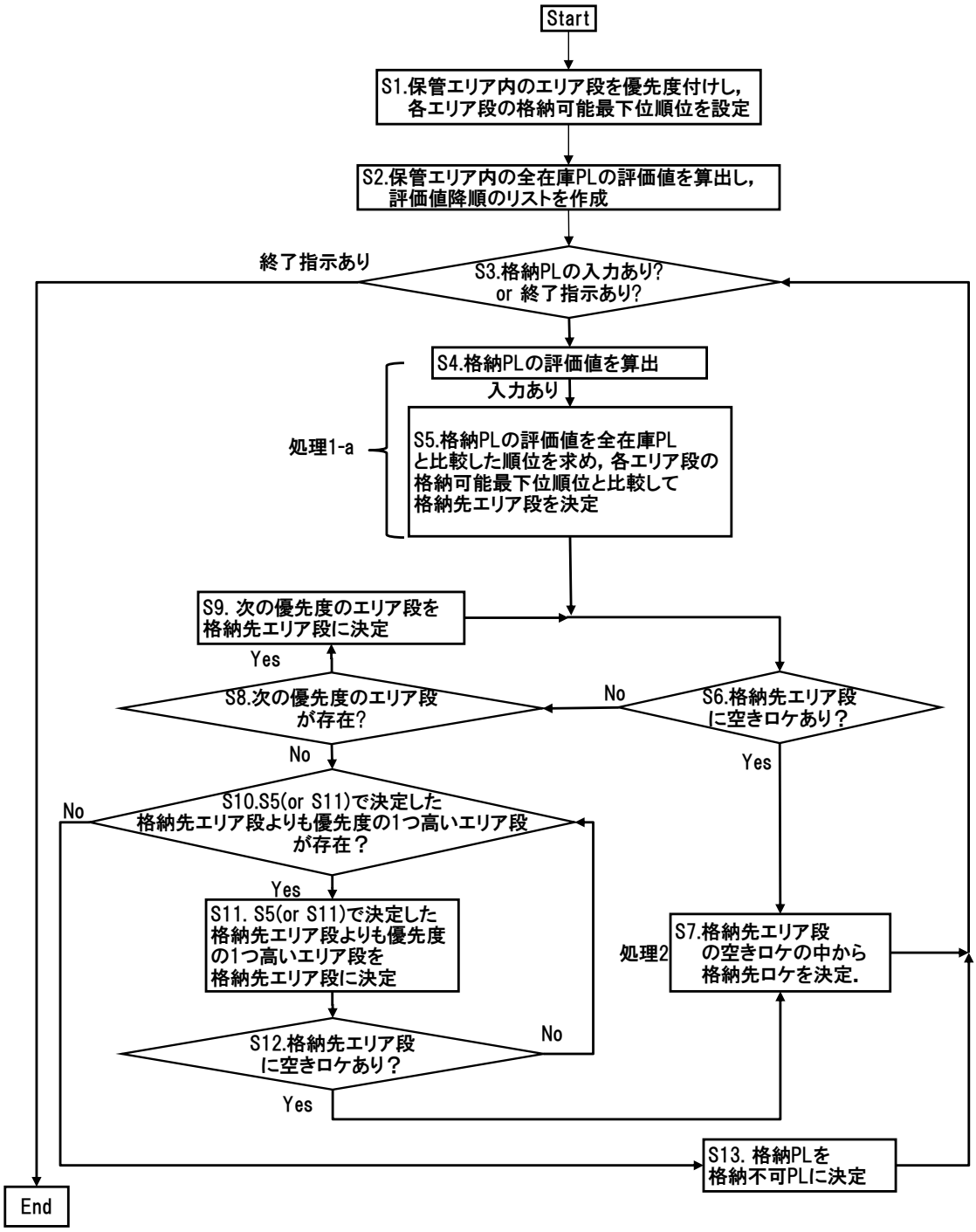

図 2-5 格納時の処理フロー

可能順位幅は各エリア段のロケ数の 0.1～1.0 倍を基準として値を設定し，それをもとに各エリア段の格納可能最下位順位を設定するのが良い．格納可能順位幅と出庫工数の関係は後述の 2.5 節で説明する．

次に S2 では，保管エリア中の在庫 PL の評価値を求め，評価値降順のリストを作成する．S3 では，格納 PL の入力があればステップ S4 に進み，終了指示があれば SPLP のシステムを終了させる．格納 PL の入力情報としては，PL 上の商品の商品名，個数，ロット順が必要であり，WMS から取得することが考えられる．

S4，S5 では 2.3.3.1 節で説明した処理 1-a に基づいて，格納 PL の評価値を求め，在庫 PL との比較から順位を求め，各エリア段の格納可能最下位順位と比較して格納先エリア段を求める．S6 では，S5 で決定した格納先エリア段に空きロケが有るかを参照し，空きロケが有れば S8 に進む．空きロケが有る場合は，S8 において処理 2 によって格納先ロケを求め，SPLP の利用者に格納先ロケを通知する．また，格納 PL を評価値リストに加え，格納先ロケを予約済に設定しておくことで，次の格納 PL にはこの格納先ロケが選ばれないようにする．

次に S6 において，S5 で決定した格納先エリア段に空きロケが無い場合は，S8 にて次の優先度のエリア段が存在するかを参照し，存在する場合は次の優先度のエリア段を格納先エリア段として設定して，再び S6 にて空きロケを探索し，空きロケが有れば S7 にて格納先ロケを決定する．空きロケが無い場合は再び S8 にて次の優先度のエリア段があるか調べ，次の優先度のエリア段が有れば再び S6 の処理に進む．

一方 S8 において次の優先度のエリア段が存在しない場合は，S10 にて，S5 で決定した格納先エリア段よりも優先度の 1 つ高いエリア段が存在するかを参照し，存在すれば S11 にて，S5 で決定した格納先エリア段よりも優先度の 1 つ高いエリア段を格納先エリア段として，S12 で空きロケがあるかを参照する．S12 で空きロケが有れば S7 に進んで格納先ロケを決定する．S12 で空きロケが無ければ，再び S10 に戻って，S11 で決定した格納先エリア段よりも優先度の 1 つ高いエリア段を格納先エリア段として，再び S12 で空きロケがあるかを参照し，空きロケが有れば S7 において格納先ロケを決定し，空きロケが無ければ再び S10 に戻る．ここで S10 において S5 もしくは S11 で決定した格納先エリア段よりも優先度の高いエリア段が存在しなければ，S13 に進み，格納 PL を格納不可 PL と定める．

2.4.2 在庫移動時の処理フロー

次に在庫移動時の SPLP の処理フローを説明する．在庫移動では評価値 E_k が優先度 1 のエリア段の格納可能最下位順位以上であれば，PL を優先度 1 エリア段に在庫移動させる．移動対象 PL を優先度 1 エリア段の最下位順位以上の PL に限定するのは，評

価値が十分に高い PL に限定することで，在庫移動を行うことで得られる出庫工数削減効果が高いと考えられる PL のみを移動対象とするためである．優先度 1 エリア段に空きロケがない場合は，優先度 2 以下で移動対象 PL が現在有るエリア段よりも優先度高のエリア段に移動させる．

在庫移動時の処理フローを図 2-6に示す．まずS1にて，予め毎日最大で何PLまで移動させるかを意味する在庫移動設定数nの値を設定しておく．ここで$n = 0$の場合には在庫移動は行われず，SPLPは格納時の格納先決定処理のみが行われる．次にS2にて，優先度2以下のエリア段内の全在庫PLの評価値E_kを算出する．その後S3にて，評価値E_kが優先度1のエリア段に格納可能な最下位順位の評価値以上の在庫PL計m件 $(m \leqq n)$ を移動対象PLと選択する．

以降の S4～S6 の処理は m 件の移動対象 PL に対して繰り返す．まず S4 にて，優先度 1 エリア段に空きロケがあるかを参照する．空きロケが有れば S5 にて移動対象 PL を優先度

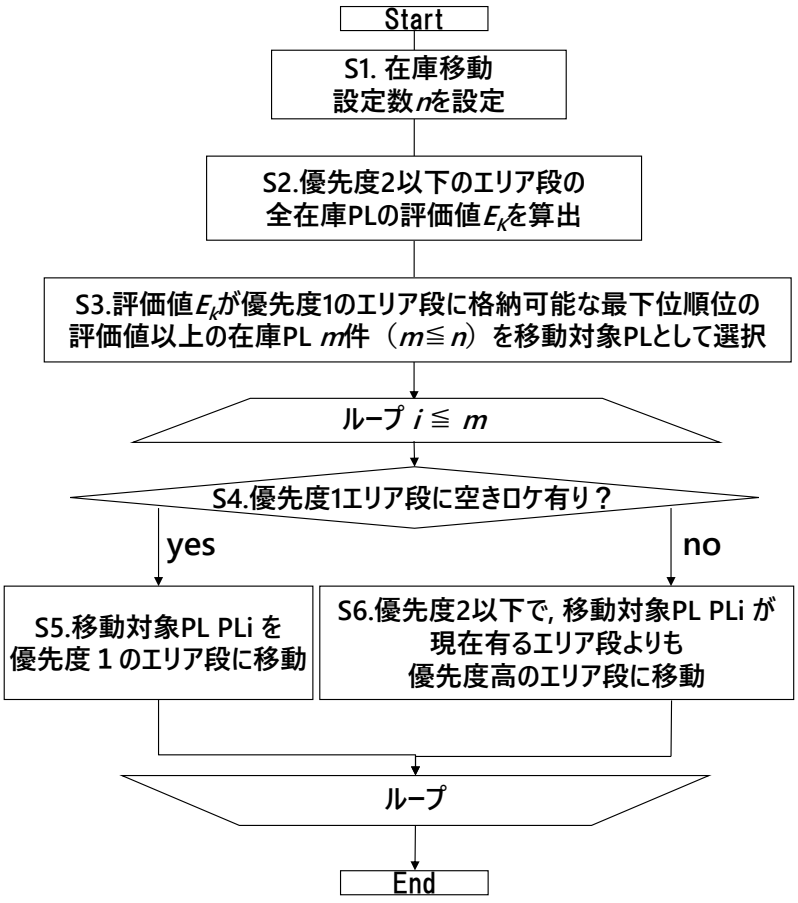

図 2-6 在庫移動時の処理フロー

1 エリア段に移動させる．また優先度 1 エリア段においては，2.3.2 節の処理 2 に記載の方法で，PL 上の各商品の出庫パタンごとの出庫タッチ数の比率から格納先ロケを決定する．

もし空きロケが無い場合は，S6 にて優先度 2 以下で，かつ現在移動対象 PL が存在するエリア段よりも優先度の高いエリア段の空きロケに移動させる．もし空きロケが存在しない場合は，在庫移動は行わない．

2.5 出庫工数削減効果の評価

本節ではシミュレーションによって出庫工数削減効果を検証し，SPLP が既存手法として PL に対する出庫タッチ数のみを評価して格納先を決定する方法に比べて出庫工数を削減できることを確認し，提案手法の有効性を示す．

2.5.1 評価方法

評価で行うシミュレーションでは 3 か月間，日々の「在庫移動 ⇒ 格納 ⇒ 出庫（CS 出庫・補充出庫）」を 1 日ずつ繰り返す．格納の方法として，既存手法と SPLP の 2 種類を評価し，3 か月間の在庫移動に掛かる総工数（総在庫移動工数）と総出庫工数を既存手法と SPLP とで比較する．既存手法，SPLP とも各エリア段に設定する格納可能順位幅の値を変えて複数回評価を行う．

既存手法では，格納先エリア段を決定する際に PL に対する出庫タッチ数のみを考慮する．すなわち，（２－５）式の PL_k の評価値を，下式（２－７）とした場合を想定する．

$$\boldsymbol{E_k = \sum_{i=1}^{n} \sum_{j=1}^{2} T_{k,i,j}} \qquad （２－７）$$

既存手法では，2.3.2 節に記載した SPLP の処理 1-a の処理が次の 1-a’のように変わり，その後処理 2 が行われる．

- 処理 1-a’：格納 PL の出庫タッチ数から格納 PL の評価値を求め，保管エリアに格納済の在庫 PL の評価値と比較した際の格納 PL の順位から格納先エリア段を決定．
- 処理 2：格納先エリア段内で，CS 出庫と補充出庫の双方の出庫出口までの運搬工数の和が最小となるロケを格納先ロケとして決定．

既存手法では，格納時の格納先決定と在庫移動を組み合わせて行う方法は行われていないことから，在庫移動は行わない．

保管エリアでは，上の全ての商品が無くなったロケが別のPLを格納，在庫移動させることができるものし，保管エリアが満杯になった時点でその日のその時点以降の格納PLは格納不可PLとみなされる．格納不可PLからの出庫は行われないようにしている．

その後出庫が進んで保管エリアに空きが発生すれば，翌日以降の格納が行われる．格納不可PLが発生するのは，データ入手元倉庫では保管エリアが満杯になりそうな時に，各ロケに残っている少量の商品を数か所のロケ（PL）に集める作業が数か月に一度行われ，これにより空きロケが数百ロケ作られるが，シミュレーションではこれを再現してないことによる．CS出庫・補充出庫はロット順に行われ，PLから商品が無くなれば，次のロットのPLからCS出庫・補充出庫が行われる．

次にシミュレーションにおける初期在庫の作成方法を説明する．初期在庫が空の状態でシミュレーションを開始すると，各エリア段に空きが多いことから，優先度高エリア段における格納，出庫が多く発生し正当な評価とならない．また本来提案手法を倉庫で長期に運用した場合は，格納期間の長いPLでも格納後の日数が経過すれば出庫タッチが掛かる．そのため，格納時の格納期間が短く出庫タッチ数が多いため優先度高エリア段に格納されたPLにも，格納期間が長く出庫タッチ数が少ないため優先度低エリア段に格納されたPLにも，一定比率で出庫タッチが発生することから，各エリア段の出庫タッチ数割合が概ね一定になると考えられる．そこで，まず初期在庫無しの状態で各エリア段からの出庫タッチ数比率が概ね一定となる2か月間，各格納方法に応じた初期在庫生成用ミュレーションを行って初期在庫を生成する．その初期在庫を用いて3か月間評価用のシミュレーションを行い，出庫工数を評価する．

表 2-2にシミュレーションで用いたデータと設定値を示す．用いたデータは，稼働中の倉庫の格納，CS 出庫，補充出庫の各実績データである．初期在庫生成後の空きロケ数は1,185である．評価値 E_k 算出に用いるCS出庫・補充出庫データの期間は2週間としたが，これは期間を長くし過ぎると季節等によって出庫タッチ数，出庫個数が変わる商品の出庫

表 2-2 シミュレーションで用いたデータと設定値

項目		値
CS出庫回数（回）		26,750
補充出庫回数（回）		25,433
格納PL数 （PL）	合計	3,340
	（うち混載PL）	（884）
初期在庫生成後の空きロケ数		1,185
シミュレーション期間		3か月間
評価値 E_k 算出に用いるCS出庫・補充出庫データの期間		2週間（14日）

タッチ数，格納期間を正しく評価できないためである．2 週間の期間が適切であるかの評価は今後の課題である．また出庫実績のない新商品は，既存商品の出荷実績の平均値で評価する．新商品は良く出荷されるものから余り出荷されないものまで様々あるが，既存商品の出庫実績の平均値で新商品を評価すれば大きくは外れないと考えたためである．新商品の適切な評価方法についても今後の課題である．

次に，総出庫工数と総在庫移動工数の算出方法を説明する．式（２－１）から，総出庫工数は,

総出庫工数（人分） ＝ 総在庫取得工数（人分）+ 総運搬工数（人分）（２－８）

と表される．ここで,

・総出庫工数：シミュレーション中の合計出庫工数

・総在庫取得工数：シミュレーション中の合計在庫取得工数

・総運搬工数：シミュレーション中の合計在庫取得工数

である．総在庫取得工数 M は下式（２－９）より算出する．

$$M = \sum_{i=1} N_{touch,i} * m_i \qquad （２－９）$$

- $N_{touch,\ i}$：優先度iのエリア段の総出庫タッチ数
- m_i：優先度iのエリア段の1出庫タッチ当たり在庫取得工数（人分）

続いて，総運搬工数の算出方法について説明する．シミュレーション中の各ロケには予め CS 出庫と補充出庫の各出庫出口までの距離（通路を加味した動線距離）を設定しておき，各ロケの CS 出庫と補充出庫の総出庫タッチ数にそれぞれの出庫出口までの距離の 2 倍（往復のため）を掛けて総運搬距離を算出し，総運搬距離をフォークリフトの平均時速（5km / h）で割って総運搬工数とする．

次に総在庫移動工数を説明する．ロケ p からロケ q まで 1 回の在庫移動をする場合，1 回の在庫移動工数は,

在庫移動工数 ={ 出庫出口（日々の初回の在庫移動時）or 1 回前の在庫移動先のロケ q’
（日々の 2 回目以降の在庫移動時）} からロケ p への移動工数
+ ロケ p からの移動対象 PL 取得工数
+ ロケ p からロケ q までの移動工数
+ ロケ q への PL 格納工数 （２－１０）

となる．ここで出庫出口は CS 出庫出口とする．これは在庫移動時には作業員は出荷バースに停車しているフォークリフトに載ってまずロケ p に向かうためである．（２－１０）式をもとにシミュレーション中に行われた全在庫移動の在庫移動工数の総和を取って総在庫移動工数とする．評価においては，既存手法（在庫移動なし）と SPLP（在庫

移動あり）の総在庫移動工数と総出庫工数との和を比較する．ここで，（２－１０）式のロケ p からの移動対象 PL 取得工数とロケ q への PL 格納工数について説明する．出庫タッチ時の在庫取得では，2.2.2 節で説明したように在庫取得後に PL を PL ラックに戻すため，フォークリフトのリフトを 2 回上げ下げする．一方在庫移動では，PL ごと運ぶためフォークリフトのリフトの上げ下げは 1 回で良い．そのため，ロケ p からの移動対象 PL 取得工数とロケ q への PL 格納工数は, 出庫タッチ時の在庫取得工数の 1/2 を設定する．

2.5.2　保管エリアの詳細と設定したパラメータ

評価に用いる倉庫データと保管エリアの詳細を説明する．図 2-7 に，図 2-2 で示した倉庫全体のレイアウトにおける保管エリアを詳細に説明したレイアウトを示す．PL ラックは 4 段構成であり，図 2-7 のように配置されている．ロケ数は合計 3,085 である．なお各ロケに 1PL ずつ格納されるため，各エリア段に格納可能な PL 数は各エリア段のロケ数と一致する．ここで対象倉庫では，日々約 100～200PL の保管エリアへの格納があり，日々扱う item 数は約 7～8 千 item である．扱う商品は化粧品やトイレタリ関係の商品（シャンプー，石鹸など）であり，倉庫で扱われる商品の大半は季節による商品の入れ替わりはなく年間を通じて同じ商品が扱われる．格納 PL の約 1/4 を混載 PL が占めており，PL 上の商品の組み合わせは作業員の判断でランダムに決定される．なお，混載 PL の商品の組み合わせをランダムとせず，格納期間や出庫タッチ数が近い商品同士を組み合わせて混載 PL を作成する方法も考えられる．しかし，混載 PL の商品の組み合わせを決定するには，各商品の個数と各商品の CS の容積情報が無いと実施困難であるが，執筆者の知る限り国内では CS の容積情報が整備されていない倉庫が多い．本

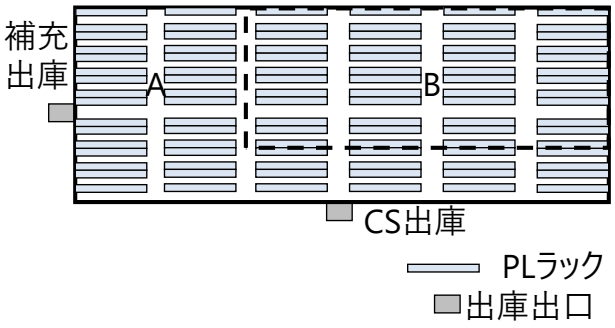

図 2-7 対象とする保管エリアのレイアウト

章で対象とした倉庫でも CS の容積情報が入手できないため，混載 PL 上の商品の組み合わせ決定は今後の課題とする．

評価におけるエリア段設定を説明する．まず，CS 出庫と補充出庫の 2 つの出庫出口に相対的に近いエリアを A，遠いエリアを B と 2 つのエリアに分割している．次に段については表 2-1 で示したように，3 段目と 4 段目の在庫取得工数がほぼ同一であることから，3,4 段目を区別なく「3, 4 段」と定義して同じ段として扱い，在庫取得工数は 4 段目の 3.0 分を設定する．以上により，エリア段は A-1 段，A-2 段，A-3,4 段，B-1 段，B-2 段，B-3,4 段の 6 種類となる．

表 2-3 に各エリア段のロケ数を示す．エリア段は，A，B エリアの何段目であるかをもとに設定する．表 2-4 にシミュレーションで設定したエリア段の優先度と在庫取得工数，平均運搬工数，平均出庫工数を示す．平均運搬工数はエリア段内の各ロケからの運搬工数の平均値を取ったものである．シミュレーションでは運搬工数は平均ではなく，各ロケから

表 2-3 各エリア段のロケ数

エリア段名	ロケ数
A-1 段	360
A-2 段	360
A-3,4 段	832
B-1 段	366
B-2 段	366
B-3,4 段	801
合計	3,085

表 2-4 各エリア段の優先度と平均出庫工数（人分）

エリア段	優先度	(a)在庫取得工数（人分）	平均運搬工数（人分）		平均出庫工数（人分）	
			(b)CS 出庫	(c)補充出庫	CS 出庫 ((a) + (b))	補充出庫 ((a) + (c))
A-1 段	1	1.9	1.1	2.3	3.0	4.2
A-2 段	2	2.3	1.1	2.3	3.4	4.6
B-1 段	3	1.9	1.9	3.1	3.8	5.0
B-2 段	4	2.3	1.9	3.1	4.2	5.4
A-3, 4 段	5	3.0	1.1	2.3	4.1	5.3
B-3, 4 段	6	3.0	1.9	3.1	4.9	6.1

の運搬工数で算出している．また平均出庫工数は，在庫取得工数と平均運搬工数の和を取ったものである．在庫取得工数は A，B エリアに関わらず，段が同一であれば同一である．

2.5.3　評価内容

シミュレーションでは，まず予備実験として各エリア段に設定する格納可能順位幅と SPLP の総出庫工数の関係を明らかにする．予備実験では在庫移動設定数を 0 とし，在庫移動は行わない．次に本番評価として，既存手法と SPLP との総出庫工数＋総在庫移動工数の比較を行い，SPLP の有効性を示す．用いる入力データと設定したパラメータは予備実験および本番評価の各格納方法で共通である．

2.5.4　評価結果

2.5.4.1 各エリア段の格納可能順位幅と SPLP の総出庫工数の関係

まず，予備実験の結果を示す．シミュレータでは，各エリア段の格納可能順位幅を各エリア段のロケ数の倍数で指定している．予備実験では，各エリア段の格納可能順位幅の値を比較的小さい値の組み合わせと比較的大きな値の組み合わせの 2 通りで設定し，総出庫工数を評価する．まず格納可能順位幅の値が比較的小さい値の組み合わせの場合に設定した各エリア段の格納可能順位幅を示す．

・A_1 段：0.1 倍，0.2 倍，0.3 倍，0.5 倍
・A_2 段：0.1 倍，0.2 倍，0.25 倍，0.3 倍
・B_1 段：0.5 倍，1.0 倍
・他：1.0 倍

優先度の高い A_1 段，A_2 段の格納可能順位幅を他のエリア段よりも小さくしたのは，予備実験の前にシミュレーションを試行した結果，優先度高エリア段の格納可能順位幅を小さくした方が総出庫工数は少なくなる傾向が分かったためである．

まず表 2-5 に各シミュレーションにおける格納可・不可 PL 数，総出庫タッチ数を示す．これらの値は毎回のシミュレーションで同じ値となる．1 回のシミュレーションに掛かる時間は約 12 秒である．図 2-8 に，在庫移動無し（在庫移動設定数 $n=0$）の場合の SPLP による総出庫工数を示す．図 2-8 において，格納可能順位幅を幅と称す．また図 2-8 の横軸はそれぞれ A_2 段，B_1 段のロケ数の何倍を格納可能順位幅として指定したかを示している．例えば「0.1_0.5」であれば A_2 段はロケ数の 0.1 倍，B_1 段は 0.5 倍を意味する．凡例の A_1 段の幅も同様に A_1 段のロケ数の何倍であるかを表す．

表 2-5 シミュレーションで発生した格納可・不可 PL 数，総出庫タッチ数

項目	値
格納可 PL 数（PL）	3,117
格納不可 PL 数（PL）	223
総出庫タッチ数（回）	26,873

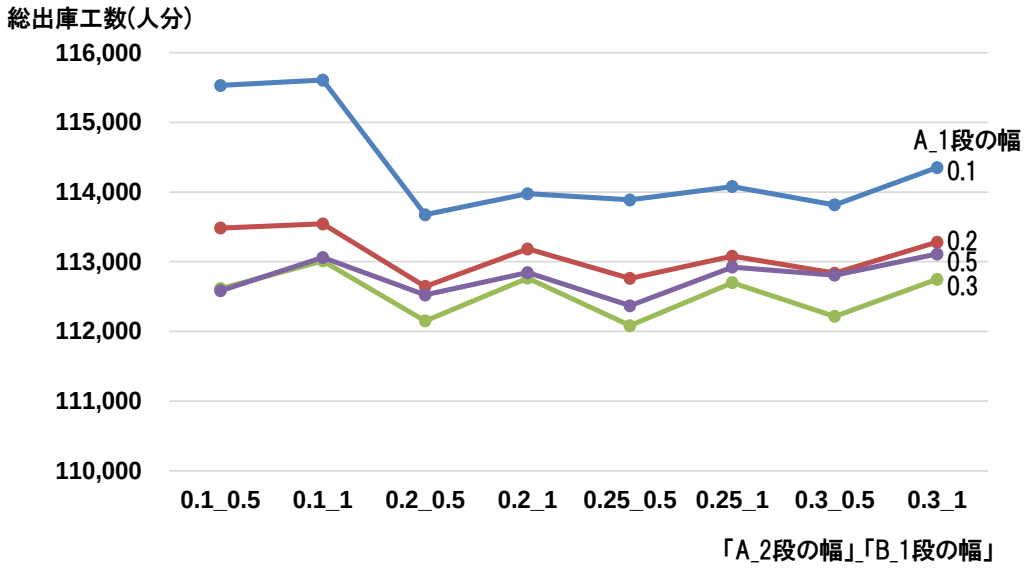

図 2-8 格納可能順位幅が小さい場合の SPLP ($n = 0$) の総出庫工数

図 2-8 を見ると，総出庫工数は A_1 段の格納可能順位幅の影響が大きいことが分かる．また A_1 段の幅をロケ数の 0.3 倍にした時が総じて総出庫工数が少なく，A_1 段の格納可能順位幅を更に狭めても広めても総出庫工数が増える傾向が有る．これは，優先度高エリア段の格納可能順位幅を狭くし過ぎると評価値の高い PL が格納されなくなり，一方格納可能順位幅を広くし過ぎると，本来格納されるべきでない評価値の低い PL が格納されてしまうことに依ると考えられる．

次に，各エリア段の格納可能順位幅を比較的大きな値を設定した場合を説明する．各エリア段に設定した格納可能順位幅は次のとおりである．

・A_1段：0.5倍，1.0倍，1.5倍
・A_2段：0.5倍，1.0倍，1.5倍，2.0倍，2.5倍
・B_1段：0.5倍，1.0倍，1.5倍，2.0倍，2.5倍
・他：1.0倍

図 2-9 にシミュレーション結果を示す．A_1 段の格納可能順位幅を広げるほど総出庫工数は増え，A_2 段以降の格納可能順位幅の大きさは殆ど総出庫工数に影響しなくなることが分かる．

今後 SPLP を業務に適用する場合は，優先度 1 のエリア段の格納可能順位幅は他のエリア段よりも小さく設定し，優先度 2，3 番目のエリア段の格納可能順位幅は優先度 1 のエリア段よりも大きな値を設定すれば良いと考えられるが，倉庫で扱う商品やエリア段の分け方等でこの結果は変わって来るため，適用先の倉庫で適宜シミュレーションを行って格納可能順位幅の値を決めれば良い．また格納可能順位幅については基本的に日々入出荷される商品の大半は同じものが扱われる中で，総出庫工数が最小となる格納可能順位幅の組み合わせは変わらないと考えられることから，毎日格納可能順位幅を見直す必要性は無い．ただし倉庫で扱う商品の大多数が変わった際や，保管エリアのレイアウトが変わった際などには適宜シミュレーションも行って格納可能順位幅を見直せば良い．

2.5.4.2 既存手法と SPLP との比較

次に，既存手法と SPLP との総出庫工数＋総在庫移動工数の比較についてを説明する．本節では予備実験の結果に基づき，A-1 段，A-2 段，B-1 段の格納可能順位幅を振ってシミュレーションを行い，既存手法，SPLP の結果とも最も総出庫工数＋総在庫移動工数が少ない結果を示す. 表 2-6 に既存手法と SPLP のシミュレーション結果を示す. 既存手法は 2.5.1

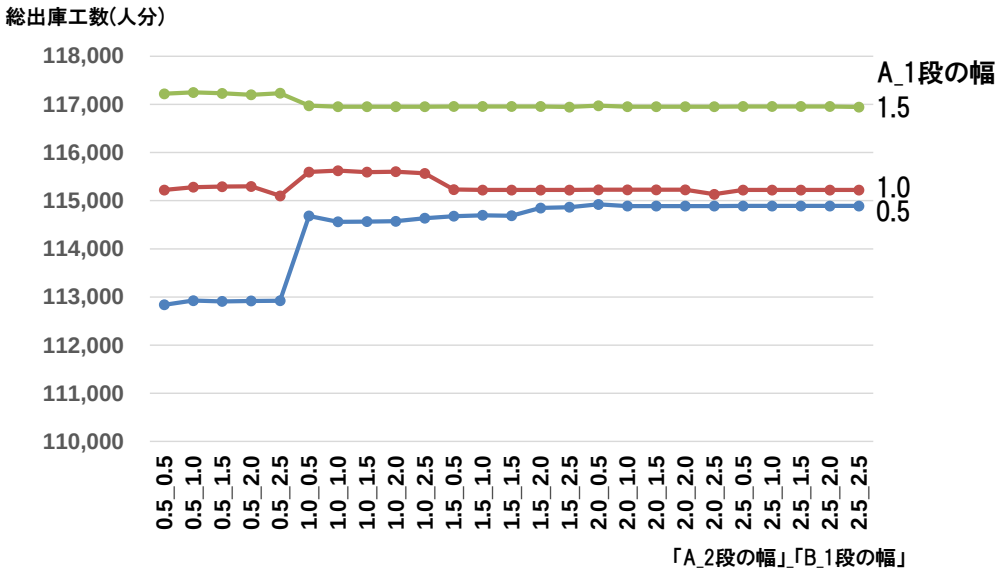

図 2-9 格納可能順位幅が大きい場合の SPLP ($n = 0$) の総出庫工数

表 2-6 既存手法と SPLP において総出庫工数＋総在庫移動工数が最小となった
シミュレーション結果

	既存手法	SPLP				
在庫移動設定数 n	0	0	5	10	20	50
在庫移動数（回）	0	0	422	638	667	618
総在庫移動工数（人分）	0	0	1,055	1,606	1,523	1594
総在庫取得工数（人分）	65,313	62,464	61,349	60,233	60,945	60,885
総運搬工数（人分）	51,926	49,619	48,991	48,776	48,549	48,139
総出庫工数（人分）	117,239	112,083	110,340	109,009	109,494	109,014
総出庫工数 ＋ 総在庫移動工数（人分）	117,239	112,083	111,395	110,615	111,160	110,618
総出庫工数＋ 総在庫移動工数削減率 (%)	0	4.40	4.98	5.65	5.19	5.64

節で述べたように，出庫タッチ数のみを用いて格納先エリア段を決定し，その後 SPLP の処理 2 と同様の方法によって格納先ロケを決定する方法である．SPLP では在庫移動設定数 n を 0 回から 1 日 50 回まで振っている．在庫移動数はシミュレーション中に行われた在庫移動の回数を表している．総出庫工数＋総在庫移動工数削減率は出庫タッチ数のみを算出して格納先エリア段を決める既存手法からの削減率を意味する．表 2-6 を見ると，SPLP の総出庫工数＋総在庫移動工数削減率は在庫移動無し ($n = 0$) の場合 4.40%，在庫移動を行った場合，$n = 10$ の時が最も総出庫工数＋総在庫移動工数削減率が大きく 5.65%となる．表 2-7 に表 2-6 に示した各シミュレーション結果における格納可能順位幅を示す．各結果において，A_1 段の格納可能順位幅は 0.5 倍以下に設定した時が総出庫工数＋総在庫移動工数が小さくなる. 図 2-10 に既存手法からの SPLP の各在庫移動数における総出庫工数＋総在庫移動工数削減率を示す.

表 2-7 表 2-6 の各結果における格納可能順位幅（×ロケ数）

	既存手法	SSLP				
在庫移動設定数 n	0	0	5	10	20	50
A_1 段	0.5	0.3	0.2	0.2	0.2	0.2
A_2 段	0.25	0.25	0.5	0.7	0.5	0.5
B_1 段	0.5	0.5	0.5	0.5	0.5	0.5
他	1	1	1	1	1	1

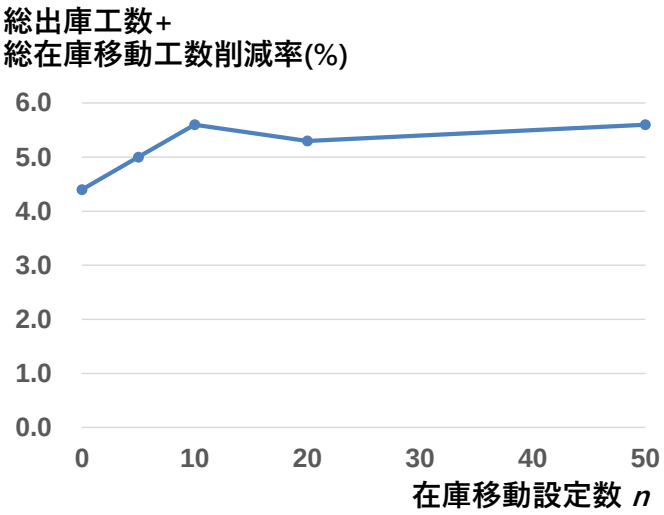

図 2-10 既存手法と比べた総出庫工数＋総在庫移動工数削減率（%）

2.6 考察

2.5.4.2 節の結果から，SPLP では出庫タッチ数のみで格納先エリア段を決める既存手法よりも総出庫工数＋総在庫移動工数を削減できる．表 2-8 に，表 2-6 で示したシミュレーション結果におけるエリア段ごとの出庫タッチ数割合を示す．SPLP の出庫タッチ数割合は在庫移動無しと総出庫工数＋総在庫移動工数が最小の $n = 10$ の結果を示す．SPLP を用いることで優先度 1 の A-1 段の出庫タッチ数割合が，既存手法と比べて在庫移動無し($n = 0$) の時は 9 ポイント，$n = 10$ の時は既存手法から 20 ポイント上昇してい

ることが分かる．このようにA_1段からの出庫タッチ割合が増加することで，1段目からの出庫が既存手法よりも増えて総在庫取得工数が削減される．さらに双方の出庫出口に近いAエリアからの出庫タッチが増えることで総運搬工数も削減されている．

次に在庫移動数 n について考察する．図 2-10 を見ると在庫移動数 n を 50 にしても総出庫工数＋総在庫移動工数削減率は変化が無く削減効果が飽和している．これは表 2-6 に記載の在庫移動数を見ると，優先度1エリア段に空きロケが足りないことから600回台（638回，667回，618回）で頭打ちとなっており，仮に在庫移動設定数 n を更に大きくしてもこれ以上在庫移動回数は増えないことから判断できる．また在庫移動設定数 n を増やすと，優先度1エリア段が在庫移動PLで占められ，評価値の高い格納PLが優先度1エリア段に格納できなくなる．そこで在庫移動数 n は，在庫移動実施前の日々の優先度1エリア段の空きロケ数から，格納PLが優先度1エリア段に日々格納される平均PL数を差し引いた値に日々動的に設定することが望ましい．本章のシミュレーションでは，在庫移動はその日の格納 PL に載る商品が決定する前に行っていることから，在庫移動の時点で優先度 1 エリア段に何 PL が格納されるかは判明しないが，在庫移動の時点で優先度 1 エリア段に格納される PL 数が判る場合には，在庫移動設定数 n は優先度 1 エリア段の空きロケ数から格納 PL 数を差し引いた値に日々動的に設定することが望ましい．これについては今後の課題である．

また更に総出庫工数＋総在庫移動工数削減効果を得るためには，優先度1エリア段に格納されたPLでも，出荷のピークを過ぎたなどの理由で評価値が下がったPLは優先度低エリア段に移動させて，優先度1エリア段の空きロケを確保することが有効と考える．また優先度1エリア段において，残りの物量が少なくなった商品を1つのロケに集約して空きロケを確保することも有効である．

表 2-8 エリア段ごとの出庫タッチ数割合（%）

	既存手法	SSLP：n = 0	SSLP：n = 10
A-1 段	17	26	37
A-2 段	14	19	15
B-1 段	15	14	14
B-2 段	15	11	8
A-3,4 段	22	19	17
B-3,4 段	17	11	9
計	100	100	100.0

本章では出庫パタンが CS 出庫と補充出庫の 2 パタンであり，出庫の際に都度商品を出庫出口に搬出する倉庫を想定して評価を行った．提案手法は出庫パタンが 2 パタンの倉庫に適用可能であるが，出庫工数削減効果は各段からの在庫取得工数，保管エリアの広さ（各ロケから出庫出口までの距離），保管エリア内のエリア段分けの仕方，優先度高エリア段の日々の空きロケ数などに応じて変わると考えられる．現時点では，他の倉庫でも SPLP によって格納期間を算出して格納先を決めることで優先度高エリア段からの出庫タッチ数割合が高まり，出庫工数削減が図れると考えるが，他の倉庫における評価は今後の課題である．また，出庫の際に作業員が複数ロケを巡回してピックする場合の対応も今後の課題である．

2.7 まとめ

格納 PL 上の各商品の出庫タッチ数と格納期間を出庫実績と在庫データから評価し，出庫タッチ数が多く格納期間の短い PL を優先度高エリア段に格納および在庫移動を行う格納先決定方式 SPLP を提案した．出庫パタンとして 2 パタンがあり，出庫の際に都度出庫出口に出庫対象商品を運搬する問題設定においてシミュレーションを行い，SPLP は格納先エリア段を出庫タッチ数のみで決定する既存手法と比べて在庫移動無しの場合に 4.40%，在庫移動を 1 日 10PL ずつ行うことで 5.65%の総出庫工数＋総在庫移動工数を削減できることを示した．

主な今後の課題として，次の 5 点が有る．1 点目については 2.6 節の最後に言及した，出庫パタンが 2 の他の倉庫に適用した場合の出庫工数削減効果の評価と，出庫パタンが 2 以外の倉庫に適用するためのアルゴリズムの拡張である．

2 点目は対象倉庫とは異なる商品を扱う倉庫への対応である．今回日々入出荷の傾向が大きく変わらない化粧品，トイレタリ商品で評価した．日々入出荷の傾向が大きく変わらない商品を扱う倉庫では過去の出庫実績から格納先を決定する方式である SPLP は適用可能と考えるが，評価は今後の課題である．そのような入出荷の傾向が大きく変わらない商品の例としては，加工食品（生鮮食品(魚・肉、野菜等)を除く食品全般）や日用品(台所・洗面・トイレ用品、文具、その他雑貨など)などが考えられる．一方で，例えば衣類のように季節によって扱う商品が入れ替わる商品の場合，直近 2 週間の出庫実績にその商品に対する出庫タッチが無ければ評価値は最低の 0 となり，優先度低エリア段に格納されてしまう．そのため，例えば前年の同じ時期の出庫実績を参照して評価値を算出するなどの対応が必要となる．

3 点目は，出庫時に作業員が複数のロケを巡回してピックする場合への対応である．複数ロケを巡回する場合でも，在庫取得工数は 1 回ごとのピックで発生するため，提案手法をそのまま適用した場合でも在庫取得工数削減効果は得られる．一方運搬工数については巡回でピックする場合，1 回の巡回で最も出庫出口に近いロケから出庫する商品が巡回経路の長さに大きな影響を与える．そのため，例えば PL の格納先決定には SPLP を用いた上で既存の巡回経路最適化手法や出庫オーダに割り当てるピック対象商品の最適化手法（例[39][40]）を組み合わせて，1 回の巡回で優先度高エリア段のみ（もしくは優先度低エリア段のみ）の巡回になるように巡回経路や出庫オーダの最適化を行うことで，巡回でピックする場合でも運搬工数削減が図れると考えるが，詳細な検討・評価が必要である．

4 点目については，本章では PL 上の商品の混載の組み合わせは作業員が判断するものとして，研究の対象外とした．しかし，格納期間や出庫タッチ数が近い商品同士を組み合わせて混載 PL を作成することで，更に出庫工数削減が図れる．そのような PL 上の混載の組み合わせ最適化に取り組む必要がある．

最後に 5 点目については，本章では格納 PL 上の商品は格納直前まで確定できない倉庫を対象とし，在庫移動は格納 PL の格納先決定より前に行うものとした．しかし格納 PL 上の商品が事前に分かる場合は，在庫移動対象 PL の移動先と格納 PL の格納先を同時に決定する方式の検討が必要である．その際に，2.6 節で述べたように，日々の在庫移動設定数 n は優先度 1 エリア段の空きロケ数から優先度 1 エリア段に格納される格納 PL 数を差し引いた値に動的に設定することが望ましい．

第3章
分割 RH 方式による BC 利用トレーサビリティ管理システム

3.1　緒言

近年，食品や医薬品を中心に SC におけるトレーサビリティ管理のニーズが高まっている[9]. トレーサビリティとは，商品の製造や原産地から販売までの履歴を記録しておき，不良品などの問題が起きた時に履歴をさかのぼって原因箇所を特定しやすくしておくこと，および商品を取り扱う作業者や一般消費者が，SC の各工程で商品が正しく品質管理されていたかを確認可能としておくことを意味する[4].

トレーサビリティを実現するために，SC の各拠点において，TR.データと呼ばれるデータを記録し，トレーサビリティ管理システムが保有するデータベース（Database，以下 DB と略す.）に記録し，利用する。TR.データは，拠点における作業内容，日時，場所，温度履歴などが含まれる．TR データの記録は SC の各拠点における商品の製造や各拠点からの入出荷，販売など，対象商品に何らかの作業が行われるタイミングで行われる. ここで TR.データが記録される契機となる各拠点の作業のことをイベントと称す. TR.データの記録については，その記録の手間を削減するためにトレーサビリティ管理の対象商品に予め RFID や QR (Quick Response) コードなどの個体識別のためのタグを取り付け，そのタグをバーコードリーダなどの端末で読み込むことで TR.データの記録がなされることが一般的である.

ここで SC においては，メーカ，卸，小売（コンビニなど）の複数企業（ステークホルダ）が関わる．そのため，TR.データは SC の上流から下流まで複数企業を跨いで共有できる必要がある．またデータの信頼性保持のため，DB に記録された TR.データは改ざんできない必要がある[45]．データを複数企業間で共有でき，データの耐改ざん性が保証されている技術として BC があり[10][11], 近年 BC を活用したトレーサビリティ管理システムが登場している[12][13]. しかし, BC を用いても依然次の 2 点が課題となる.

1 つ目は，TR.データの秘匿性である．本章で扱う秘匿性は，TR.データ内の商品の情報を閲覧権限のあるステークホルダのみに開示されることと定義する．TR.データの国際標準である EPC の TR.データでは，商品が梱包箱等に梱包されると，1 つの TR.デー

タに梱包された商品の情報（商品 ID）が記載される．例えば物流倉庫で複数商品の同一の梱包箱への梱包（混載）された場合，ある商品の TR.データの開示要求元が自社商品を追跡する場合に，TR.データに含まれる他社商品の情報は秘匿される必要がある．これは，仮に他社商品の情報を開示要求元が閲覧できると，開示要求元が倉庫からの出荷先である店舗に対する他社商品の出荷数・出荷時期などが把握できてしまい，店舗の経営・販売戦略等が開示要求元に流出する恐れがあるためである．

2 つ目は，BC の書き込み性能である．経済産業省は，「コンビニ電子タグ 1,000 億宣言[14]」を公表しており，年間 1,000 億個のコンビニ商品に電子タグを取り付けることが想定されている．仮にそれら商品の TR.データを BC で取り扱う場合，店舗だけでも約 3,600 Tx (transaction) / sec の書き込み性能が求められる．一方で既存の BC 基盤の書き込み性能は，数百 Tx / sec 程度に留まり[16]，高速な TR.データの書き込みを実現する施策が必要である．

以上の課題を解決するため，本章では TR.データの BC の書き込み性能とデータの秘匿性を両立する書き込み手法を用いたトレーサビリティ管理システムを提案する．提案の書き込み手法では，複数の TR.データのハッシュ値をまとめたルートハッシュ（RH）を BC に書き込むことで，書き込み回数削減による書き込み性能の担保を図る．また，梱包箱に混載された商品の TR.データの情報を製造元メーカごとに分割した後に RH を生成する方式である分割 RH 方式を提案し，データの秘匿性を保証する．

なお，BC を用いた場合でも BC 書き込み前の入力データの改ざんを防ぐことは困難であるが，これについては本章の検討の対象外とする．ただし，分割 RH 方式では TR.データをそのまま BC に書き込むよりも BC への書き込み回数が減る（書き込む間隔が長くなる）ため，TR.データをそのまま BC に書き込むよりも入力データに不正ができる時間が生じる懸念が有る．そのため，分割 RH 方式適用時に入力データに不正ができる時間を短くする書き込みタイミングの検討は本章の対象とする．

以降，3.2 節では BC 書き込み時の課題，3.3 節では提案手法の分割 RH 方式の内容，3.4 節では秘匿性とコストの評価，3.5 節では書き込み性能の評価，3.6 節ではデータの検索と改ざん有無の検証時間の評価，3.7 節でまとめについて述べる．

3.2　BC 書き込み時の課題

3.2.1　トレーサビリティ管理の概要

本章では，RFID もしくは QR コードによる個体識別タグが取り付け可能な物品をトレーサビリティ管理の対象とする．例えば食品，日用品などが挙げられる．図 3-1 に本章で対象とするトレーサビリティ管理サービスの構成を示す．ここでトレーサビリティ管理サービスは TR.データを活用したサービスを意味し，トレーサビリティ管理システムはトレーサビリティ管理サービスを実施するために TR.データを取り扱う情報システムを意味する．トレーサビリティ管理サービスにはサービス提供者とサービス利用者である各企業の作業者，一般消費者がステークホルダとして関与し，サービス提供者，各企業の拠点，一般消費者の間でネットワークを介して TR.データが共有される．ここで図 3-1 の企業 3 に店舗が複数存在するように，各企業には複数の拠点が存在することがある．トレーサビリティ管理サービスには，例えば一般消費者が店舗などで商品購入前にスマートフォンなどの端末で個体識別タグを読み込み，流通過程の温度管理履歴などを参照するトレースバックがある．また各拠点の作業者もトレースバックの利用者であり，作業者の拠点に入荷された商品の製造から入荷までの過程の温度管理履歴等を参照することがある．またトレーサビリティ管理サービスの別の例として，メーカなどの各企業が出荷済商品のリコール対応時に，対象商品の出荷先等を調べるトレースフォワードがある．

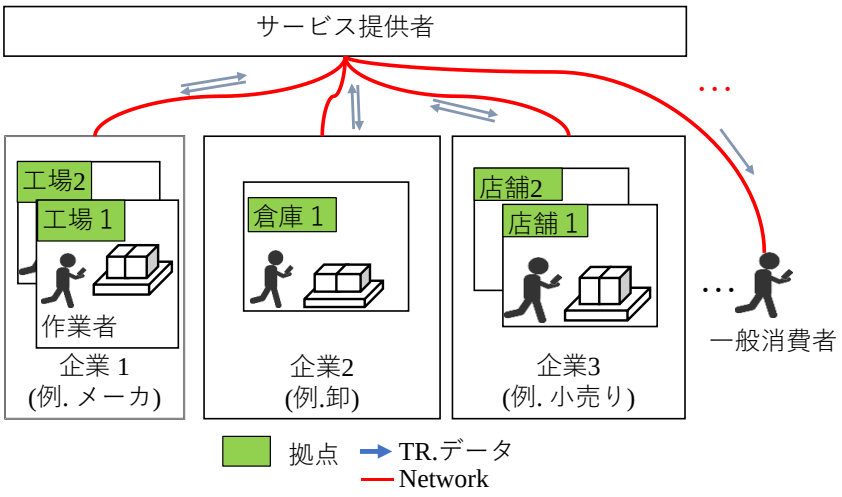

図 3-1 トレーサビリティ管理サービスの構成

BC を活用したトレーサビリティ管理システムでは，複数企業の各拠点間が BC Network でつながれ，各拠点において TR.データを BC に書き込む．書き込みは各企業の作業者が行い，商品の個体識別タグをバーコードリーダなどで読み込むことで，TR.データが書き込まれる．一方 TR.データの閲覧は一般消費者および作業者が個体識別タグを端末で読み込んで行う. サービス提供者は各企業から取得した TR.データを管理し，一般消費者および作業者からの閲覧要求に基づいて TR.データを開示する．

次に TR.データを記録するための商品の識別方法を説明する．TR.データは国際標準の EPC に基づくデータを想定する．個々の商品には個体識別タグが取り付けられ，EPC の GS1 (General Specifications 1) 識別コードである GTIN (Global Trade Item Number) [41] により一意に識別される. 各物品は流通の過程で梱包箱に梱包されることが一般的である．その場合，梱包箱にも識別タグが付けられ，一意に識別される．

ここで EPC では，1 つの TR.データは 1 つの XML ファイルとして取り扱われる．本章では EPC の TR.データが記載された XML ファイルを TR.データファイルと称す. TR.データファイルの内容について説明する．ここでは EPC の用語に合わせて商品を物品と称す. 図 3-2 に, メーカ A でワイン a1～a6 が Case1 の梱包箱に梱包される場合の TR.データファイルの例を示す．1 行目にイベント種別（図では Aggregation Event：物品の集約・梱包の意味）が記載され，2 行目以降に eventTime：発生日時，parentID：親 ID，childEPCs：子 ID，bizStep：イベント名，readPoint：発生場所が記載される．ここで親 ID と子 ID については，商品が梱包箱に梱包されている場合，親 ID には梱包箱の物品 ID，子 ID には梱包箱の中の物品の物品 ID が記載され，商品が梱包箱に梱包されてない場合は，親 ID のみ物品 ID が記載される．物品 ID の記述には前述の GTIN が用いられ，「物品の製造元企業 ID.物品 ID」の形で記載される．

```
<AggregationEvent>
  <eventTime>2019/6/24 10:20</eventTime>
  <parentID>MakerA.Case1</parentID>
  <childEPCs>
      <epc>MakerA.Wine a1</epc>
      <epc>MakerA.Wine a2</epc>
       …
      <epc>MakerA.Wine a6</epc>
  </childEPCs>
  <bizStep>Packing</bizStep>
  <readPoint>MakerA</readPoint>
</AggregationEvent>
```

図 3-2 TR.データファイルの例

3.2.2　発生するトレーサビリティデータ

コンビニ電子タグ 1,000 億宣言実現時に発生する TR.データの種類と発生件数を説明する．表 3-1 に，メーカ，倉庫（卸の企業に所属），店舗（小売り企業に所属）で発生する TR.データの種別と発生件数を示す．メーカでは商品が製造されて Case と呼ばれる梱包箱に梱包後，PL に載せられて倉庫に出荷される．ここでコンビニでは年間 1,000 億商品が扱われることから，年間 1,000 億個の商品が製造されるものとし，メーカの製造における秒間の Tx 数は，1,000 億個を 365（日）* 24（時間）* 3,600（秒）で割って，3,171Tx / sec となる．その他のイベントにおける TR.データの発生件数は，1 梱包箱には商品が 20 個，1PL には梱包箱が 10 箱載ると仮定して算出している．

以降倉庫における TR.データについて説明するが，混載の割合等の数値は倉庫作業員へのヒアリングに基づく．倉庫では，PL から梱包箱が降ろされた後，全梱包箱の約 1/4

表 3-1 各々の拠点で発生するイベントと TR.データの秒間 Tx 数

拠点	NO	イベント	年間 Tx 数	秒間 Tx 数	合計秒間 Tx 数
メーカ	1	製造	1,000 億	3,171	
	2	梱包箱に梱包	50 億	159	
	3	検品	50 億	159	
	4	PL に乗せる	5 億	16	
	5	PL 出荷	5 億	16	3,521
倉庫	1	PL 入荷	5 億	16	
	2	PL から降ろす	5 億	16	
	3	開梱※	13 億	40	
	4	オリコンに詰め替え※	13 億	40	
	5	かご車乗せ	13 億	40	
	6	出荷	13 億	40	192
店舗	1	入荷	13 億	40	
	2	かご車から降ろす	13 億	40	
	3	検品	50 億	160	
	4	開梱	50 億	160	
	5	販売	1,000 億	3,171	3,571
合計					7,284

※オリコンで混載する場合のみ発生．全体の梱包箱の 1/4 と仮定

に平均で 3 社程度の商品の混載が発生する．ここで混載される商品は Case の梱包箱を開梱して取り出された後に，オリコン（折り畳みコンテナの略）と呼ばれる専用の梱包箱に新たに詰め込まれる．続いて，混載されていない梱包箱である Case と，混載された梱包箱であるオリコンがかご車に載せられて店舗に出荷される．かご車には梱包箱（Case とオリコンの双方）を平均 4 つ載せる．Case（混載無し）とオリコン（混載有り）の割合は 3：1 である．また，かご車に載せる梱包箱（Case）の中身は混載されていないが，一つのかご車に入荷元が異なる複数メーカから入荷した Case の梱包箱を載せるケースもある．その場合も一つのかご車に載る Case の梱包箱の入荷元メーカ数は 3 程度である．図 3-3 にかご車とオリコンの外観を示す．その後，店舗にて入荷後かご車から降ろされ，検品，開梱を経て販売される．秒間の Tx 数は，表 3-1 に示す通り店舗で 3,571Tx となる．

3.2.3　BC 書き込み時の課題

トレーサビリティ管理システムに BC を活用することによってデータの耐改ざん性が保証されるが，依然次の 2 点が課題となる．

・課題(1) TR.データの秘匿性

・課題(2) BC の書き込み性能

課題(1) について説明する．TR.データを各企業の作業者が閲覧する際に，自社に入荷されなかった商品の TR.データは秘匿されるべきである．この例を，ある商品の製造から店舗への出荷の流れを示している図 3-4 を用いて説明する．a, b 社はメーカ，A～C 社は卸，x, y, z は小売の各企業と仮定する．図中の点線は梱包を表し，梱包される商品のメーカが異なる場合は混載を表す．

図 3-3 かご車（左）とオリコン（右）の外観

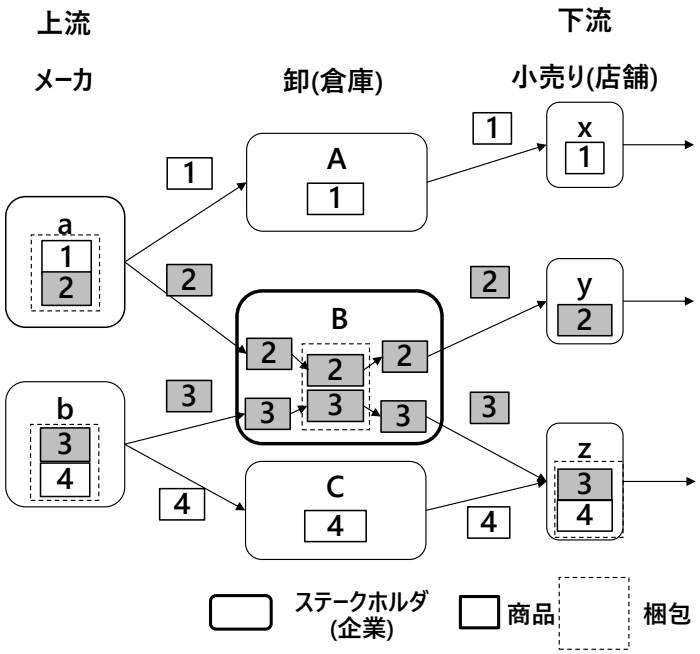

図 3-4 データ秘匿性の課題

この例では，B 社においてメーカの異なる商品 2 と商品 3 が同一の梱包箱に一時的に混載され，その後商品 2 は y 社，商品 3 は z 社に出荷されている．B 社が商品 2, 3 の TR.データを閲覧する場合，B 社に入荷されなかった a 社の商品 1，および b 社の商品 4 の TR.データを閲覧できないようにしなければならない．これは，仮に閲覧できてしまうと，上流の a, b 社の取引先企業や商品の出荷数・出荷時期などが B 社に開示され，a, b 社の経営・販売戦略等が不必要に流出する恐れがあるためである．また，下流の z 社の商品 4 の TR.データも同じ理由で閲覧できないようにする必要がある．図 3-4 においては，B 社は色付きの商品 2, 3 の TR.データのみ閲覧可能である．また，仮に a 社が自社で製造した商品 1, 2 のデータを閲覧する場合，B 社において同一の梱包箱に混載された商品 3 の情報も同様の理由で閲覧できない．しかし，EPC では混載が発生すると，図 3-2 に示す TR.データファイルの childEPCs の欄に複数の商品名が記録されるため対策が必要である．また閲覧者が一般消費者の場合でも，閲覧対象商品が SC の過程で他の商品と混載されていた場合には，一般消費者に各企業で取り扱った他の商品の情報が不必要に閲覧されないように，複数商品が混載された TR.データファイルにおいて閲覧対象商品以外の商品の商品名（物品 ID）は秘匿する必要がある．

ここでデータの秘匿性の確保手段として，既存手法ではデータの書き込み者，閲覧者をグループ分けする方法がある．例えば BC 基盤の Hyperledger Fabric[15]では，BC 上の

データの共有範囲をチャネルという機能を使って制御することができる．しかし，この方法では予め出荷先が判明している状況であればアクセス者をグループ分けすることは可能であるが，本章で対象とする SC の過程で不特定多数の企業が参画し，出荷先も事前に分からない状況に対応困難である．

次に課題(2) について説明する．コンビニ電子タグ 1,000 億宣言に基づいて，仮に年間 1,000 億商品の TR.データを管理する場合，表 3-1 に示したように店舗における販売の TR.データだけでも全国で合計約 3,600Tx / sec の書き込みが発生する．ここで BC 基盤には管理者を有するか否かでコンソーシアム型とパブリック型に大別されるが，コンソーシアム型のほうが書き込み処理は高速である[16]．そこで本章では BC 基盤にコンソーシアム型の Hyperledger Fabric を採用するが，v1.4 における書き込み性能は数百 Tx / sec に留まり[16]，高速な書き込み処理を実現するための施策が必要である．

書き込み処理を高速化する既存手法として，マークル木 (Mercle Tree) [71] の考え方を導入して，BC に書き込み対象のデータを書き込む代わりに，複数の書き込み対象のデータ各々からハッシュ値を生成し，それら複数のハッシュ値から RH を生成して BC に書き込むことで書き込み回数を削減する方法がある[72]．マークル木では，ツリー構造の Root に対応する RH が，Leave(s)に対応する各データのハッシュから生成され，各データのいずれかが変われば，RH も変わる性質が有るため，ハッシュの代わりに BC に RH を書き込めば，各データの耐改ざん性が担保される．

図 3-5 にマークル木を用いた BC の書き込み方法例を示す．複数のハッシュ値から RH を生成することで，BC 書き込み回数を削減する．この方法を TR.データに採用した場合，TR.データは DB に保存しておき，TR.データを開示する際には，BC に格納された RH と DB から取得した TR.データから生成した RH を比較することで，改ざんがされていないことを検証する．しかしこの方法を単に用いるだけでは，混載の TR.データ

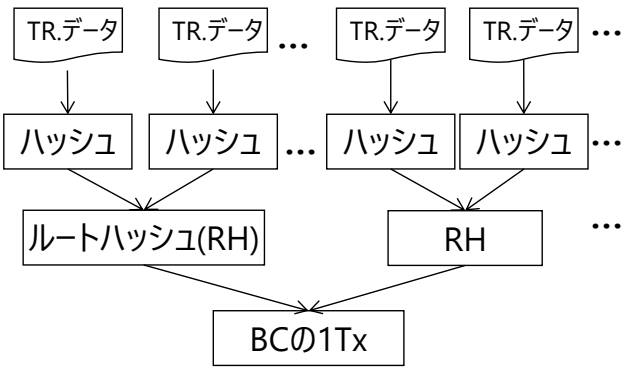

図 3-5 マークル木を用いた BC への書き込み方法

があった場合, DB にそのまま TR.データが記録されているため閲覧者が閲覧不可な TR.データを閲覧できてしまう.

3.3 分割 RH 方式

3.3.1 秘匿要件と書き込み方法の方針

表 3-2 に TR.データの秘匿要件をまとめる．アクターとして，閲覧者（各企業の作業者もしくは一般消費者）とサービス提供者が居る．まず各企業の作業者（以降，作業者）に対する秘匿要件を考える．TR.データの種類には，作業者から見て上流と下流のデータ，および作業者自身が行った作業履歴のデータ（作業者の属する企業における TR.データ）がある．3.2.3 節で述べたように，上流のデータは作業者の所属企業に対して出荷されたデータのみ, 下流のデータは所属企業から出荷したデータのみが閲覧できるようにする．また作業者の所属企業で発生した TR.データは閲覧して良い.

次に一般消費者に対する秘匿要件を考える．一般消費者は SC の最も下流に位置付けられるため，上流の TR.データを閲覧することになる．一般消費者に対しては，SC の過程で混載された商品の情報が不必要に流出しないようにする目的で, 閲覧しようとする商品の TR.データのみを閲覧できるようにする.

最後にサービス提供者に対する秘匿要件を考える．サービス提供者が仮に TR.データの内容を閲覧できると，そこに記録されている各拠点における詳細な製造・物流の履歴や各社の商品の仕入れ，販売戦略などの情報が流出する懸念がある．そのため，サービス提供者は TR.データの内容を閲覧できないようにする.

表 3-2 データの秘匿要件

アクター		データ種別	閲覧可能なデータ
閲覧者	各企業の作業者	上流の TR.データ	作業者の所属企業に出荷されたデータのみ
		作業者の属する企業における TR.データ	閲覧可能
		下流の TR.データ	作業者の所属企業から出荷されたデータのみ
	一般消費者	上流の TR.データ	閲覧しようとしている商品のデータのみ
サービス提供者		全て	閲覧不可

そこで TR.データの書き込み方法の方針として, SC の各企業の拠点で保持する DB で保存し，TR.データの RH を BC に書き込むものとする．これにより，サービス提供者は TR.データを閲覧不可となり，各閲覧者も後述の方法によってのみ閲覧可能なデータを閲覧できるようになる．

3.3.2 提案の分割 RH 方式

閲覧者に対する秘匿性の要件を満たすために, 提案手法では以下で説明するように複数製造元の商品が混載されたTR.データは，製造元メーカごとにデータを分けてDBに保存し, RHを生成する際には同一メーカの商品のTR.データから生成する．このように製造元メーカごとに元のTR.データを分割して生成したRHを分割RHと称す．また分割RHは各拠点の出荷・販売のタイミングでBCに書き込むものとする．ここでTR.データを分割する際にデータを改ざんできる余地があるが, これについては対象外としているBC書き込み前の改ざんに含まれることから，今後対策を検討する．

図 3-6 を用いて，分割 RH 方式を用いた BC への書き込み方法を説明する．ある拠点から，A 社と B 社の商品が混載された梱包箱が出荷されるとする．この時，元の TR.データには Child EPCs の欄に，製造元の A 社と B 社の商品名が記載されている．この場合，同拠点における該当商品の TR.データを集め，A 社商品のみと B 社商品のみのデータになるよう各拠点内の各 TR.データを分割する（図 3-6 中（a)）．具体的には，各 TR.データをその TR.データに含まれるメーカの数だけ複製し，各 TR.データに一社のメーカになるように他メーカの商品名は伏せ字に置換する．例えば A 社用 TR.データの場合，B 社の商品名を伏せ字とする．なお他メーカの商品名を伏せ字にする以外の方法として，最初から TR.データをメーカごとに分けて生成する方法や他メーカの商品情報を伏せ字にせず他メーカの商品名を削除する方法も考えられるが, 伏せ字にする以外の方法を用いた場合の評価については今後の課題である．例えば最初から TR.データをメーカごとに分けて生成する場合，TR.データを分割して伏せ字に置換する処理が不要となる反面，混載が発生した拠点の作業者が TR.データを閲覧する際に，メーカごとに分割した TR.データから複数商品が混載された TR.データを生成する必要がある．また他メーカの商品名を削除する方法では, 伏せ字に置換する方法と比べて分割 RH 生成時の処理の重さは変わらないが，メーカがその TR.データを閲覧する際に，混載がなされていたかどうかが直感的に分かりにくくなる課題がある．

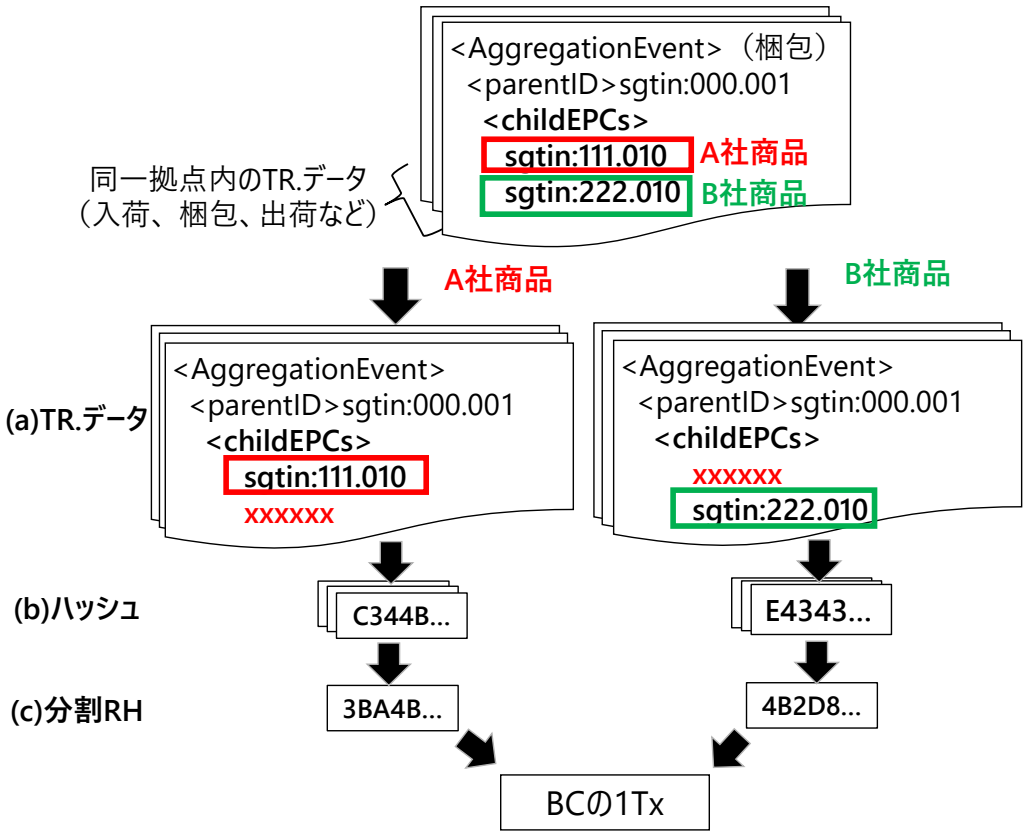

図 3-6 分割 RH を用いた TR.データの書き込み方法

次に，A 社と B 社に分割したそれぞれの全 TR.データのハッシュ値を生成し（図 3-6 中（b)），さらにそれぞれに対して分割 RH を生成する（図 3-6 中（c)）．その後 A 社用と B 社用のそれぞれの分割 RH を BC に書き込む．ここで BC に分割 RH を書き込む際は，BC に 1Tx に複数の複数の分割 RH を書き込む．このように分割 RH を生成し，1Tx に複数の分割 RH を書き込むことで BC の書き込み回数を削減する．

ここで，分割 RH を生成するタイミングを各拠点の出荷・販売時とする理由を説明する．一般に，1 つの分割 RH にまとめるハッシュ数を増やす方が BC への書き込み回数を削減できる．書き込み回数を最も削減するには，各商品が店舗で販売されるタイミングで，メーカにおける製造から店舗における販売までの全 TR.データのハッシュを 1 つにまとめた分割 RH を生成すれば良い．しかしこの場合，商品が店舗で販売されるまで分割 RH が BC に一度も書き込まれず，それまで TR.データの耐改ざん性が保証されない．また，仮に拠点を跨いだ分割 RH を生成する場合，一度サービス提供者にデータを集める必要があり，データの秘匿性が損なわれるとともに，複数の拠点から TR.データを集めるためのトランザクションが発生する．そこで，分割 RH は拠点ごとに生成する．

またその条件の下で，書き込み回数を削減する目的で，各拠点からの出荷・販売のタイミングで各拠点における TR.データの分割 RH を BC に書き込むものとする．この場合，各拠点における TR.データの耐改ざん性は各拠点から出荷・販売されたタイミングで保証される．但し，各拠点で長期に渡って出荷・販売されない商品については，BC に分割 RH を書き込むまでの日数の上限を決めておき，そのタイミングで分割 RH を BC に書き込むことが耐改ざん性確保の点で望ましい．そのような長期で出荷・販売されない商品の分割 RH の BC への書き込みタイミングの詳細化は今後の課題である．

以下，分割 RH 生成処理について説明する．ハッシュ関数を H とする．ある拠点において，出荷・販売される梱包箱もしくは商品に含まれる各商品の製造・入荷から出荷・販売までの TR.データにおいて，子 ID に記載される各商品の製造元メーカ数を $M(\geqq 1)$ とする．分割 RH は子 ID の各商品をメーカ $i(1 \leqq i \leqq M)$ ごとに分類して取得する．また各 TR.データにおいて，メーカが i 以外の商品の商品名（商品 ID）は伏せ字に置換する．ある拠点 X においてメーカ i の各商品に生じる総イベント数が N_i の時，メーカ i の各商品に生じる TR.データを TR.データ $_{i,j}$ として，メーカ i ごとに生成される分割 RH を分割 RH_{Xi} とすると，

$$\text{分割 } RH_{Xi} = H\left\{\sum_{j=1}^{N_i} H\left(\text{TR.データ}_{i,j}\right)\right\} \qquad (3-1)$$

と表される．ここで，Σ はハッシュ値をハイフン等で文字列結合することを意味する．

なお，分割 RH を生成する際に，同一拠点内の該当商品の全 TR.データを検索する必要がある．これについては，同一商品の含まれた TR.データファイル名を時系列で並べた関連情報を各拠点が別途保持することで検索を可能とする[69]．この検索に掛かる時間の評価は今後の課題である．

3.3.3 システム構成とデータモデル

前節で述べた分割RH方式を用いたトレーサビリティ管理システムの構成を図 3-7に示す．サービス提供者と SC 上の企業が BC Network でつながっている．また各企業は TR.データを保存する DB を保持し，DB 間もネットワークでつながる．BC への書き込み処理では，各拠点における出荷・販売のタイミングで分割 RH 生成が行われ，分割 RH が BC に書き込まれる．

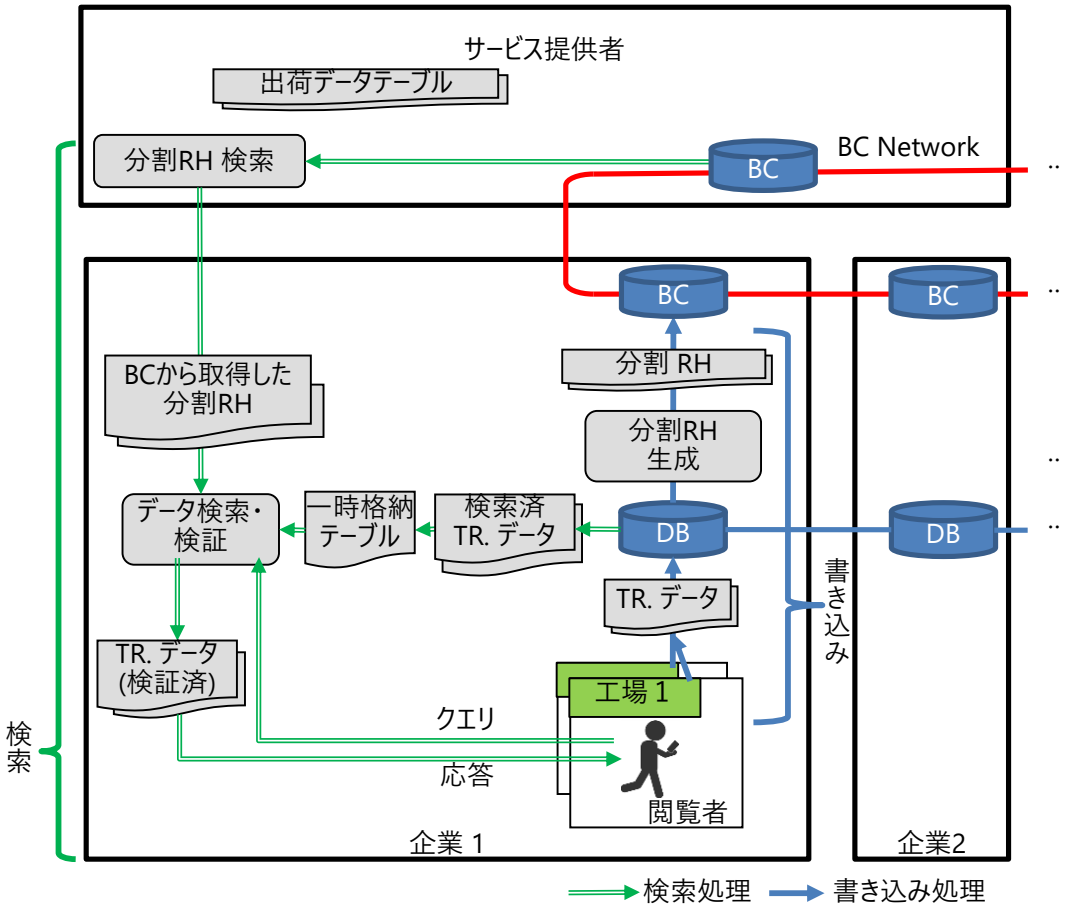

図 3-7 提案のシステム構成

ここで図 3-7 では，各企業に一つ DB と BC 型 DB（図 3-7 では BC と表記）を保持する想定で記載しているが，それらを各企業の拠点ごとに一つずつ持たせる構成でも良い．BC 型 DB は Hyperledger Fabric を採用する．これは，同基盤が他の BC 基盤に比べて書き込み処理が高速であることと，Hyperledger Fabric では key-value 型の State DB[73]を活用することで，検索時の BC からの分割 RH の取得を容易とすることができるためである．ここでサービス提供者が持つ出荷データテーブルは，TR.データ開示時に TR.データの検索を行うために保持している．詳細は 3.3.4 節にて説明する．

次に BC に書き込む分割 RH のデータモデルについて説明する．図 3-8 は BC のデータモデルを模擬した図である．Key は State DB のキーであり，検索処理のしやすさを考慮して「BC クライアント名（GLN コード）_日付_連番」とする．ここで，GLN（Global Location Number[42]）は拠点を一意に表す EPC で定められたコードである．一方 Record（Value）には，複数の分割 RH を連想配列で書き込む．その連想配列の Key は，出荷ファイル名（Shipment File Name）とし，Value として同一拠点の出荷・販売以外の TR.デ

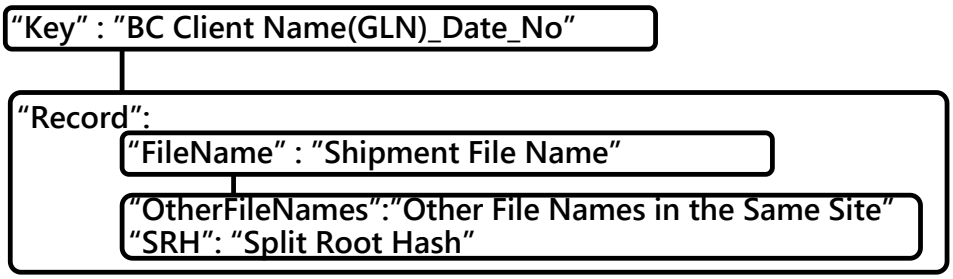

図 3-8 BC におけるデータモデル

ータ名の文字列結合と，分割 RH の値を持つものとする．出荷ファイル名についても検索のしやすさを考慮し，「商品 ID_入荷元拠点 ID」とする．

3.3.4 トレーサビリティデータの検索処理と真正性検証処理

次に TR.データの検索処理と真正性検証（改ざんがなされてないことの検証）処理について説明する．まず検索処理について説明する．閲覧者が検索要求（クエリ）を行うと，サービス提供者上で分割 RH 検索が行われ，出荷データテーブルを参照し，BC に問い合わせて BC 上の分割 RH を取得し，合わせて TR.データ名を取得する．続いて，各クライアント DB から該当する TR.データ（検索済 TR.データと称す）を取得し，TR.データの検証処理を行う．

検証処理では，検索済 TR.データから分割 RH を生成し，これを BC から取得した分割 RH と比較して，同一であることを確認する．その後閲覧者に TR.データを開示する．TR.データは店舗などで商品を購入する一般消費者も閲覧可能である．その場合モバイル端末などを用いて TR.データを検索し，同様に BC 上の分割 RH との比較を行って，改ざんがされてないことを検証する．この場合一般消費者は購入前に TR.データを検索する．そこで，店舗においては商品を陳列する際に一度分割 RH を生成し，販売後に改めて分割 RH を生成（更新）する．

次に，検索および検証処理の詳細手順を説明する．図 3-7 に示すように，検索を容易とするために，サービス提供者は各商品および梱包箱（コンテナ）の各拠点の出荷イベントに対応した TR.データ名を保持した出荷データテーブルを持つ. 図 3-9 と表 3-3 にそれぞれ，出荷データテーブルのデータモデルと出荷データテーブルの例を示す.

出荷データテーブルは商品 ID (Product ID) が Key である． 1 つの Key にその商品の各拠点の出荷データ (Shipment Data) をリストで複数持つ. 出荷データは要素として BC

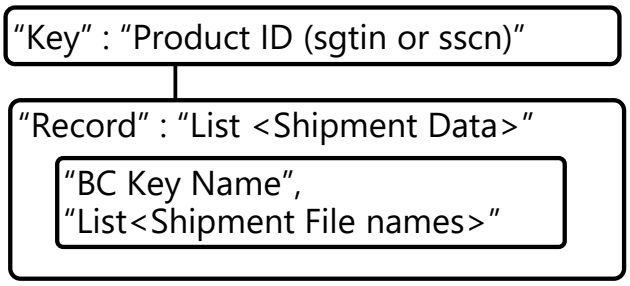

図 3-9 出荷データテーブルのデータモデル

Key 名，出荷ファイル名（1 回の出荷で分割 RH 生成が行われる場合は複数）を持つ．出荷データは時系列で並ぶ．

表 3-3 に示した例では，SGTIN1 の商品は時系列順に GLN1 と GLN3 の拠点から出荷されたことを意味する．また BC Key 名は図 3-9 の Key に該当する．ここで，SGTIN (Serialized GTIN) は個々の商品を一意に識別するコードであり，SSCC (Serial Shipping Container Code) はパレットやかご車, 梱包箱などのコンテナを一意に識別するコードである[42]．SGTIN と SSCC では，「製造元（コンテナ所有元）企業の GLN コード.製造番号」の形式で表される．ここで，表 3-3 において梱包箱である SSCC1 のデータの 1 行目には 3 つの出荷データファイル名が記されている．これは SSCC1 に製造元企業が異なる 3 社 (Supplyer GLN1～3) の商品が混載されており，3 つの分割 RH が生成されていることを意味する. なお, 出荷データテーブルは商品が梱包されて出荷された場合は, 梱包箱の出荷履歴が記録される．

図 3-10 を用いて，TR.データの検索および真正性検証処理の手順を説明する．Loop1 の中の処理は検索対象商品が辿ってきた拠点 (Site) の数だけ繰り返す．まず，S1 でク

表 3-3 出荷データテーブルの例

Product ID	出荷データ	
	BC Key 名	出荷データファイル名
SGTIN1	GLN1_Date_No	SGTIN1_SupplierGLN.xml
	GLN3_Date_No	SGTIN1_SupplierGLN.xml
SSCC1	GLN2_Date_No	SSCC1_SupplierGLN1.xml, SSCC1_SupplierGLN2.xml, SSCC1_SupplierGLN3.xml
	…	…
…	…	…

ライアントにて検索対象商品の Product ID（EPC の SGTIN または SSCC）を検索キーと

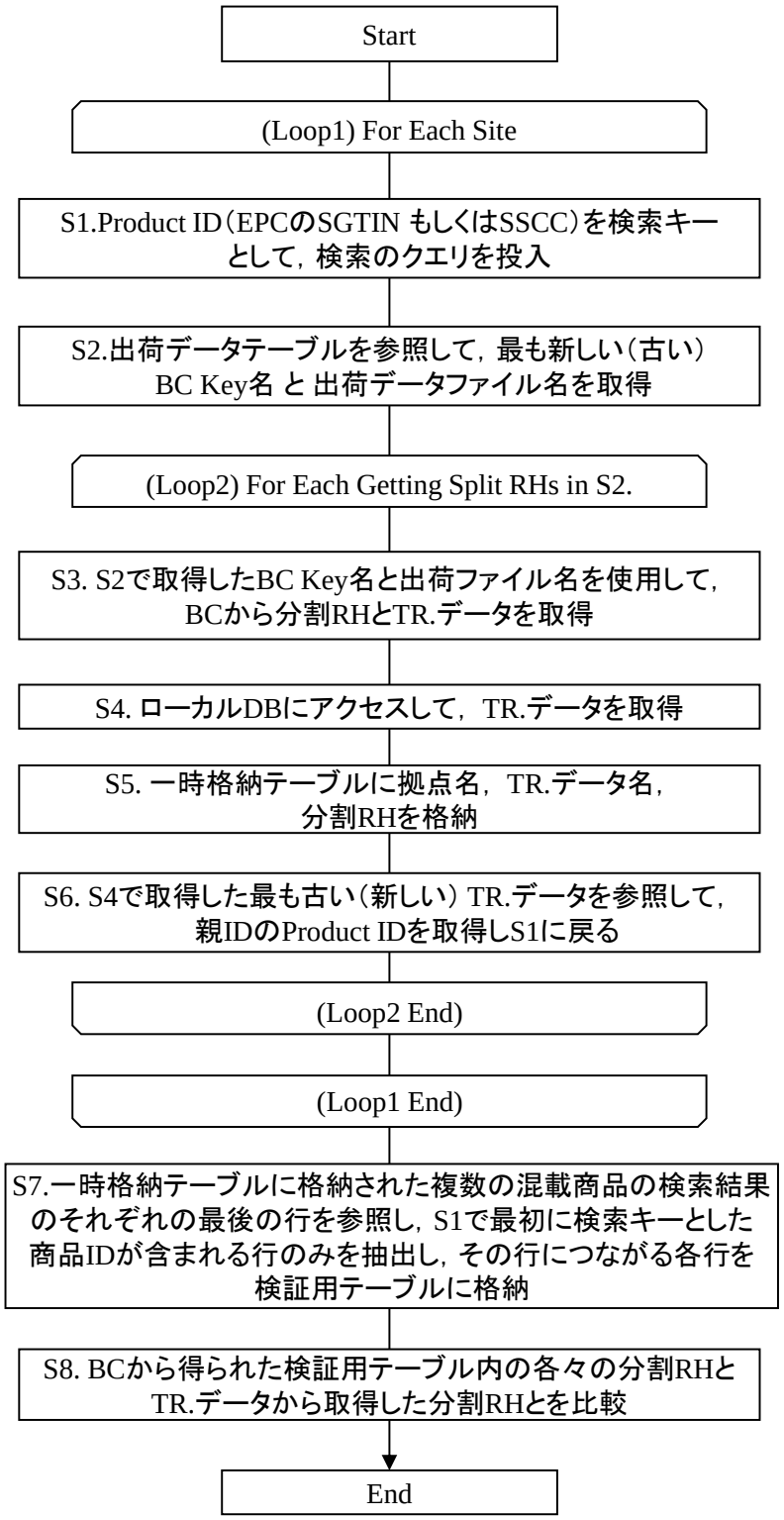

図 3-10 データの検索と真正性検証の処理フロー

して，検索のクエリを投入する．次にトレースバックの場合，S2 でサービス提供者側にて出荷データテーブルを参照し，検索対象商品の出荷データの中でその一度の検索の中で検索済でない最も新しいデータ（BC の Key 名と出荷データファイル名）を取得する．

例えば表 3-3 に示す出荷データテーブルで SGTIN1 を検索する場合，SGTIN1 の中の 2 行目のデータが該当する．また，この行の検索が終わったことは検索のアプリにてフラグなどで管理しておき，その一回の検索の Loop1 の処理で再度 SGTIN1 の行を参照する際には SGTIN1 の中の未検索の 1 行目のデータが参照されるようにする．トレースフォワードの場合は，一度の検索の中で検索済でない最も古いデータを取得する．ここォワードの場合は，一度の検索の中で検索済でない最も古いデータを取得する．ここでトレースバックとトレースフォワードの区別は，例えば検索者が利用する端末でどちらの機能を使うかを選択する．また，検索の起点となる拠点を選べるようにしても良い．

以降，Loop2 の中の S3～S6 の処理は，S2 の結果複数の出荷データファイルが検索された場合（ある拠点からの出荷時に複数メーカの商品が混載されており，分割 RH が複数生成された場合に該当），分割 RH の数だけ実施する．S3 では，サービス提供者が S2 で BC から取得した BC の Key 名をもとに分割 RH とその拠点の他の TR.データ名を取得する．

続いて S4 で，S3 で BC から取得した TR.データ名と S2 で取得した出荷データファイル名をサービス提供者からクライアント端末に送信し，クライアント端末側で各拠点の DB から TR.データを取得する．ここでどの拠点の DB から TR.データを取得するかは，S2 で取得した BC の Key 名に含まれている拠点名を参照すれば良い．

次に S5 で，S3 で取得した拠点名，TR.データ名，および BC から取得した分割 RH を一時格納テーブルに格納する．次に S6 にて S4 で取得した TR.データの中で，トレースバックの場合は時系列的に最も古いデータから（トレースフォワードの場合は最も新しいデータから）Parent ID を取得し，その Parent ID を Product ID として S1 の処理に戻り，再度 S1～S6 の処理を繰り返す．このようにして全ての拠点に対して Loop1 で S1～S6 の処理を行い，全拠点の TR.データの検索を行う．

次に，S7 以降の処理を説明する．S7 は Loop1 の処理の結果，いずれかの拠点で混載がある場合に検索対象以外の商品の TR.データも検索結果に含まれるため行う．S7 では，一時格納テーブルに格納された複数の混載商品の検索結果のそれぞれの最後の行を参照し，S1 で最初に検索キーとした商品 ID が含まれる行のみを抽出する．そして，その行とつながる一時格納テーブルの各行を検索結果とみなして，それらを検証用テーブルに格納する．最後に S8 では，検証用テーブルに格納された各行に対して，BC 上の分

割 RH と，各行の TR.データから生成する分割 RH との比較を行う．比較結果が全て一致すれば，検証結果は OK として検索要求者に TR.データを開示する．一致しない行があった場合には，サービス提供者から各拠点の利用者に TR.データの改ざんの疑いがあることを連絡する．

ここで，各拠点で管理する TR.データについては他社のデータでも検索要求をすれば誰でも閲覧できる懸念があるため，自社で取り扱ったことのない商品は検索できないようにする必要がある．そこで，検索要求をする閲覧者に対し予め拠点名とユーザ名でユーザ認証を掛けておき，図 3-10 の S1 の処理の際にその拠点名に紐づく拠点 ID（GLN コード）と，出荷データテーブルの検索対象商品の BC Key 名（BC クライアント名（GLN コード）_日付_連番）に含まれる GLN コードが一致（複数ある場合はいずれかが一致すれば良いものとする）した場合のみ検索が実行できるものとする．それ以外の時にはアクセス制御を行うなどして他拠点の DB は参照できないようにしておく．なお BC クライアントを複数拠点でまとめて 1 つ設置する場合は，例えば出荷データファイル名には出荷が実際になされた拠点 ID をファイル名の先頭に含め，その拠点 ID と検索者の拠点 ID との照合を行うことが考えられる．

またサービス提供者に関しても，各拠点の DB のアクセス制御にて DB にはアクセスできないようにするとともに，各 DB にはアクセスの証跡を記録して，各クライアントからの検索要求以外の不正なアクセスを追跡できるようにする．なお，一般消費者が TR.データを検索する場合には，一般消費者のモバイル端末からの検索要求をサービス提供者が Web サーバで受信し，検索対象商品の製造元メーカのクライアント端末に検索要求を転送して検索を行い，Web サーバを介してその結果を一般消費者に開示する．その際の通信は SSL などの既存の技術を用いてサービス提供者には見えないようにして，サービス提供者の TR.データの閲覧を防止する．また検索要求を行った商品が SC の上流過程で別商品（同一メーカを含む）と混載されていた場合，別商品の TR.データは伏せ字にするなどして，不必要に一般消費者に SC の上流過程の情報が流出することを防ぐ想定である．

3.3.5 検索処理と真正性検証処理の具体例

本節では，具体例を用いて検索・検証処理の手順を説明する．図 3-11 に，GLN が 1～3 の 3 つのメーカの商品 (SGTIN : 1.1，SGTIN : 2.1，SGTIN : 3.1) が GLN : 4 の倉庫で混載されて GLN : 5 の店舗に出荷され，SGTIN : 1.1 の商品が販売前に開梱 (Unpacking)

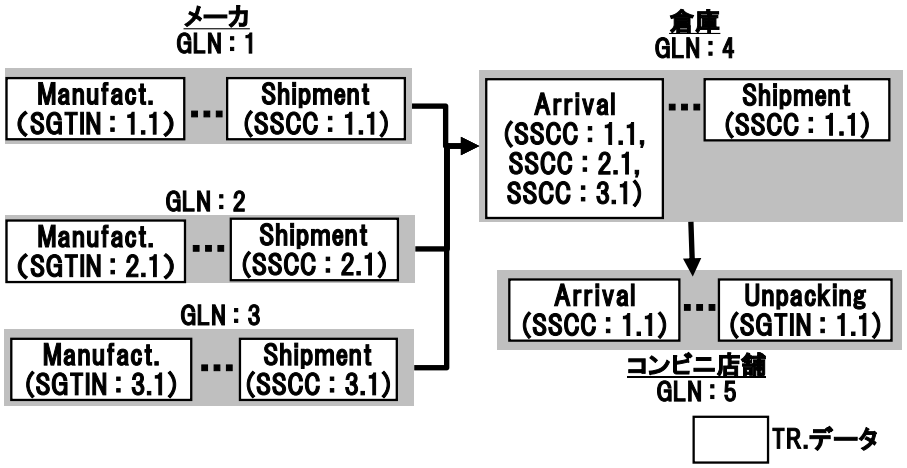

図 3-11 検索と検証を行う TR.データの例

されて陳列される例を示す．各メーカでは各商品はそれぞれ SSCC：1.1，2.1，3.1 の梱包箱に梱包されるものとし，倉庫では，SSCC : 1.1 の梱包箱にこの 3 つのメーカの商品が混載されるものとする．

この時に出荷データテーブルに記録されるデータを表 3-4 に示す．簡単のため，BC Key 名の日付は省略している．ヘッダの行を除いた 1 行目には GLN : 5 の店舗でこの商品が陳列される情報が記載されている．2 行目には SSCC : 1.1 の箱がメーカ 1 から出荷された情報が，3～5 行目には倉庫から出荷された情報がそれぞれ記載される．ここで倉庫からの出荷では 3 つのメーカの異なる商品が混載されているため，分割 RH を 3 つ生成している．そのため倉庫からの出荷では 3 行の出荷データが記載される．6, 7 行目

表 3-4 出荷データテーブルに記録される内容

Product ID	出荷データ	
	BC Key 名	出荷ファイル名
SGTIN:1.1	5_1	SGTIN:1.1-GLN4.xml
SSCC:1.1	1_1	SSCC:1.1-GLN1.xml
	4_1	SSCC:1.1-GLN1.xml, SSCC:1.1-GLN2.xml, SSCC:1.1-GLN3.xml
SSCC:2.1	2_1	SSCC:2.1-GLN2.xml
SSCC:3.1	3_1	SSCC:3.1-GLN3.xml

には倉庫で混載された商品に対応した SSCC : 2.1, 3.1 の梱包箱の出荷情報が記載されている.

今店舗で一般消費者がこの商品を購入する直前に，トレースバックを行う場合を考える．一般消費者が自身のモバイル端末等でこの商品の EPC コードを参照し，TR.データの検索を行う．表 3-5 に，図 3-10 に示した手順に従って S1～S6 の処理を行うことで生成される一時格納テーブルの内容を示す.

一時格納テーブルは検索の結果取得した各拠点のデータ（BC Key 名，拠点名，BC 上の分割 RH，TR.データの名称）が 1 行ずつ記載される．TR.データの列には各拠点の入荷（または製造）から出荷までの TR.データ名が記録される．また，「順序」と「次」の列の番号は対応しており，トレースバック，フォワードを行った際に，ある拠点の TR.データの検索後に参照する次の拠点の情報が，「次」の列に記載の番号の行に記載される．表 3-5 の例では，順序が 1 の行の次の欄に 2-1，2-2，2-3 の 3 つが記載されている．これは，倉庫で 3 つのメーカの商品が混載されており，倉庫出荷時に分割 RH が 3 つ生成されているためである．2-1 の行は検索対象の SGTIN : 1.1 の商品の倉庫における TR.

表 3-5 一時格納テーブルの内容

順序	BC key	拠点（GLN）	BC 上の分割 RH	TR. データ	次
1	5_1	5	dfehgfiowe	SSCC:1.1-1.xml, …, SGTIN:1.1-GLN4.xml	2-1, 2-2, 2-3
2-1	4_1	4	dfefiowe	SSCC:1.1-1.xml, …, SSCC:1.1-GLN1.xml	3-1
2-2	4_1	4	dfegfioe	SSCC:2.1-1.xml, …, SSCC:1.1-GLN2.xml	3-2
2-3	4_1	4	fioehrtt5	SSCC:3.1-1.xml, …, SSCC:1.1-GLN3.xml	3-3
3-1	1_1	1	dfeawse	SGTIN:1.1-1.xml, …, SSCC:1.1-GLN1.xml	-
3-2	2_1	2	daw05se	SGTIN:2.1-1.xml, …, SSCC:2.1-GLN2.xml	-
3-3	3_1	3	feawgrt	SGTIN:3.1-1.xml, …, SSCC:3.1-GLN3.xml	-

データの情報であり，2-2 と 2-3 の行はそれぞれ別メーカの SGTIN : 2.1，SGTIN : 3.1 の商品の倉庫の情報となる．

次に「順序」が 3-1～3-3 の行はそれぞれ，GLN : 1～3 の 3 つのメーカの TR.データに対応している．このように，図 3-10 の S1 から S6 までの処理では，一時格納テーブルに検索対象でない混載された別メーカの商品も含まれるため，図 3-10 の S7 の処理を行い検索対象の商品の情報のみに絞り込む．

表 3-6 に図 3-10 の S7，S8 の処理で生成される検証用テーブルの内容を示す．まず S7 の処理にて，一時格納テーブルの 3-1, 2, 3 の行を対象に商品 ID が SGTIN:1.1 の商品の TR.データが含まれる行の検索が行われる．表 3-5 では，3-1 の行の TR.データに SGTIN : 1-1.xml が含まれているため，3-1 の行が選択される．また，順序が 1，2-1 の行も 3-1 の行につながるため検索結果として選択され，検証用テーブルに記載される．

次に S8 の処理を行い，TR.データから生成した分割 RH と，BC から取得した分割 RH との比較検証が行われる．両方の値が一致すれば TR.データの改ざんが無いとみなし，検索要求者に TR.データの情報を開示する．

3.4　秘匿性とコストの評価

提案手法による秘匿性とシステムのコストを，他の BC への記録方法との比較により評価する．他の BC への記録方法として，(a) TR.データをそのまま BC に書き込む方法と，(b) ハッシュを書き込む方法，(c) メーカごとに分割していない RH を書き込む方法がある．(b)，(c) の方法では複数ステークホルダが含まれた場合でも，TR.データの分割は行わないものとする．

表 3-6 検証用テーブルの内容

順序	拠点	BC 上の分割 RH	TR.データ	TR.データから生成した分割 RH	OK/NG
1	5	dfehgfiowe	SSCC:1.1-1.xml, …, SGTIN:1.1-GLN4.xml	dfehgfiowe	OK
2-1	4	dfefiowe	SSCC:1.1-1.xml, …, SSCC:1.1-GLN1.xml	dfefiowe	OK
3-1	1	dfeawse	SGTIN:1.1-1.xml, …, SSCC:1.1-GLN1.xml	dfeawse	OK

表 3-7 に評価結果を示す．（a）の方法では BC 上に TR.データを書き込むため，誰でもデータを閲覧可能である．また（b），（c）の方法ではサービス提供者によるデータの閲覧を防ぐことはできるものの，閲覧者へのデータ開示時に閲覧者以外の物品の情報を開示してしまう．一方提案手法（d）では，いずれの秘匿要件も満足できる．

なお（b），（c）の方法では，データ閲覧者へのデータ開示の直前に，閲覧者に関連する物品情報のみを抽出して開示する方法も考えられる．しかしこの場合，開示時に行う TR.データのハッシュ（または RH）と BC に格納されたハッシュ（または RH）の比較による真正性検証は，メーカでデータを分割していないハッシュや RH 同士で比較することになり，その後 TR.データをステークホルダで分けて開示することになる．その際に，クライアント（データ閲覧者）にて TR.データを改ざんできる上に，閲覧不可な他社の商品情報を閲覧できる余地がある．そのため，分割 RH 方式のほうが耐改ざん性とデータ秘匿性が優れている．

次にコスト面での評価を行う．同じく表 3-7 に初期環境構築コスト，運用開始後の BC データの維持コスト，DB データの維持コストを示す．BC データの維持コスト，DB データの維持コストはそれぞれ DB および BC に書き込まれるデータ量に応じて従量課金制でコストが増える場合を想定する．TR.データの維持コストは DB に TR.データを保

表 3-7 データ秘匿性とコストの評価

	(a) TR.データを書き込み	(b) ハッシュを書き込み	(c) RH を書き込み	(d) 分割 RH を書き込み
サービス提供者へのデータ秘匿	×	○	○	○
閲覧者へのデータ秘匿	×	×	×	○
初期環境構築コスト	低 (BC)	高 (BC+ Local DB+NW)	高 (BC+ Local DB+NW)	高 (BC+ Local DB+NW)
BC データの維持コスト	大 (TR.データ)	低 (ハッシュ)	低 (ルートハッシュ)	中 (分割 RH)
DB データの維持コスト	無し	中	中	大 ((b)(c)の 1.1 倍)

持する場合のコストを意味する．初期環境コストについては，（b）～（d）の方式では BC に加えて，TR.データを保持・通信する DB とネットワークが必要なため，方式（a）に比べて割高となる．なお，（b）～（d）の間で差はない．

次に BC のデータ維持コストについて考察する．仮にデータ量に応じてコストが増大するとした場合，（a）は BC に直接 TR.データを書き込むため，BC への書き込み量が最大であり，BC サーバの拡張などに掛かるコストが最大となる．方式（b）はハッシュを書き込むため，BC への書き込み量は 4 方式の中で 2 番目に多く，方式（c）は分割をしない RH のため最小となる．提案方式（d）では分割 RH のため，データ容量は（c）に比べて多くなるが 4 方式の中では 2 番目に少ない．

DB データの維持コストについては，提案方式（d）では混載が発生した場合（EPC の TR.データの ChildEPCs に複数メーカの商品が記載された場合）に，メーカごとに分割した TR.データを保持することから，4 方式では最大となる．しかしコンビニ商品の場合，混載が発生するのは SC の途中の倉庫であり，倉庫の一度の出荷で 3～4 社の商品の混載と考えられる．一方，他のメーカや店舗では混載は発生しない．この場合の TR.データの増大量を見積もる．コンビニ商品の流通過程で，3.2.2 節の表 3-1 と同じイベントが発生するとし，3.2.2 節と同様 1 つの梱包箱に 20 個，1PL に 10 箱，1 かご車に 4 箱入るケースを想定する．倉庫全体の 1/4 の梱包箱に対して混載が発生した場合，EPC の TR.データの ChildEPCs に複数メーカの商品が記載されるのは倉庫の No4 の「オリコンに詰め替え」と店舗の No4 の「開梱」である．ひと箱につきメーカ 3 社の商品が混載されている場合，倉庫 No4 の TR.データの TX 量は 3 倍，店舗 No4 の Tx データは 1/4 が混載されていることから，

160 * 1 / 4 * 3 + 160 * (1 – 1 / 4) * 1 = 240 Tx / sec

発生することになる．表 3-8 に分割 RH 方式適用時の TR.データと分割 RH の秒間 Tx 数を示す．TR.データに着目すると前述の倉庫の No 4 と店舗の No 4 のイベントは分割 RH 方式適用時に増大するが，他のイベントでは TR.データの増大はなく，方式適用時には適用前に比べて 1.02 倍の TR.データの Tx 量となる．また，仮にすべての梱包箱で 4 メーカの商品が混載され，かつ 1 つのかご車に載せられる 4 つの梱包箱の中の商品が全て異なる場合でも同様に計算すると分割 RH による TR.データ件数は 1.1 倍と見積もられる．このように提案方式（d）は DB のデータの件数が 4 つの中で最も多いが，高々 1.1 倍でありかつ DB は BC に比べればデータ維持コストは少ないため，大きな問題にはならないと考える．その一方で提案方式は BC のデータ維持コストは 4 方式の中で 2 番目に少なく，データ秘匿性が最も高い点が優位である．

表 3-8 分割 RH 方式適用時の TR.データと分割 RH の秒間 Tx 数

拠点	NO	イベント	合計 TR.データ 秒間 Tx 数	分割 RH 適用時 TR データ 秒間 Tx 数	分割 RH 適用時 分割 RH 秒間 Tx 数
メーカ	1	製造	3,171	3,171	16
	2	梱包箱に梱包	159	159	
	3	検品	159	159	
	4	PL に乗せる	16	16	
	5	PL 出荷	16	16	
倉庫	1	PL 入荷	16	16	120
	2	PL から降ろす	16	16	
	3	開梱※	40	40	
	4	オリコンに詰め替え※	40	120	
	5	かご車乗せ	40	40	
	6	出荷	40	40	
店舗	1	入荷	40	40	3,171
	2	かご車から降ろす	40	40	
	3	検品	160	160	
	4	開梱	160	240	
	5	販売	3,171	3,171	
合計			7,283	7,443 (1.02 倍)	3,307 (0.45 倍)

※倉庫全体の梱包箱の 1/4 が混載されると仮定

次に分割 RH 数については，メーカと倉庫では出荷のタイミングで分割 RH を取得することで，TR.データを BC に書き込む場合に比べて Tx 数を削減できる．店舗では最後の販売のタイミングで分割 RH を生成すると秒間 3,171Tx となり，分割 RH 生成を行わない場合の秒間 3,571Tx に比べて若干の削減となる．今後店舗の分割 RH 生成時の書き込み件数を削減するには，例えば販売のタイミング分割 RH を生成する際に，その 1 つ前の作業の 4.開梱で同じ梱包箱に梱包されていた商品が全て販売したらそれら商品で 1 つ分割 RH を生成することで，秒間 240Tx の書き込み件数に削減できる．これについては今後詳細検討を行う．

なお提案手法では，流通過程で発生する混載回数が増えるに従い TR.データと分割 RH 数が増大する課題が有る．コンビニ商品では混載が発生するのは倉庫のみであり，混載も 1 回か多くて 2 回しか発生しないため TR.データと分割 RH 数の増大は大きな問

題にならないが，提案手法を他の分野に適用する場合は対策が必要であり，今後の課題である．例えば自動車などの製造業の Supply Chain で，部品のトレーサビリティを管理する場合，複数回に渡りコンビニ商品の混載に該当する部品の組み立てが行われ，分割 RH 数は増大する．

最後に，（a）～（d）の各方式に共通する課題として，参加企業が増えた場合の BC のスケーラビリティがある．これについては，まず Hyperledger Fabric では v1.0 以降でノードの動的な追加を可能としており[75]，参加企業増に柔軟に対応できる．一方でノード増に伴い，ノード間でコンセンサスを取るための処理が性能ボトルネックとなる懸念がある．これについては，例えばコンセンサスアルゴリズムを全ノードの合意ではなく代表者（例：各企業の本社のノードのみの合意とする等）の合意として Tx を削減することが対策として考えられ，また，扱う商品の分類や製造元企業の地域などで BC のチャネルを分けてデータを管理することで，コンセンサスの対象を同一チャネル内に絞り込むことが Tx の削減に有効と考えられるが，評価については今後の課題である．

3.5　書き込み性能の評価

本節では，BC の書き込み性能の評価について述べる．

3.5.1　評価目的と内容

表 3-8 を見ると，分割 RH 生成時でも BC への書き込みは店舗で 3,171Tx / sec，全拠点の合計で 3,307Tx / sec 発生する．本節では，コンビニの SC を模擬した実験環境を構築し，BC の 1Tx に複数の分割 RH を書き込むことで，これら件数の分割 RH が書き込めることを検証する．

3.5.2　書き込み性能の評価モデル

単位時間あたりに BC に書き込み可能な分割 RH 数 N_{RH}（件 / sec）は，

・N_t：BC の Tx のスループット（Tx / sec）

・n_r：BC の 1Tx に書き込む分割 RH 数（件 / Tx）

とすると，

$$N_{RH} = N_t \times n_r \qquad （3－2）$$

となる．そのため，n_r をパラメータとして変えた場合の N_t を計測する．ここでハッシュ関数は SHA-256[74]を用いる．

3.5.3　BC 書き込み性能の評価手順

図 3-12 に示すクラウドの評価環境を用いて，BC の書き込み性能の評価を行う．評価では，1 社あたり 1 拠点と仮定し，コンビニ商品を取り扱うメーカを 3 拠点（3 社），倉庫 1 拠点，店舗 1 拠点の計 5 拠点として BC の書き込み性能の計測を行う．DB と BC Client は各拠点に 1 つとしてそれぞれ計 5 つとし，BC Server は 2 台とする．Peer は各サーバ 1 つずつの計 2 つ，Orderer は片方の BC サーバに 1 つ設置し，Endorsement Policy は「and」とする．ここで Peer は Hyperledger Fabric において BC の DB である State DB へのデータアクセスを行うエンティティであり，Orderer はデータアクセスの順番を制御するエンティティである．Peer が State DB にデータを書き込む際には各クライアントからの書き込み要求を Peer 間で共有し，各 Peer で予め保持している State DB へのデータ書き込みルールであるチェーンコードをシミュレーション実行（これを Endosement と称す）し，各ピアの Endosement の結果が書き込みルールを満たしていれば，State DB への書き込みが行われる．ここで Endosement Policy は複数の Peer が有る時に，and を設定すると and で結ばれた両方の Peer の Endosement の結果が書き込みルールを満たせ

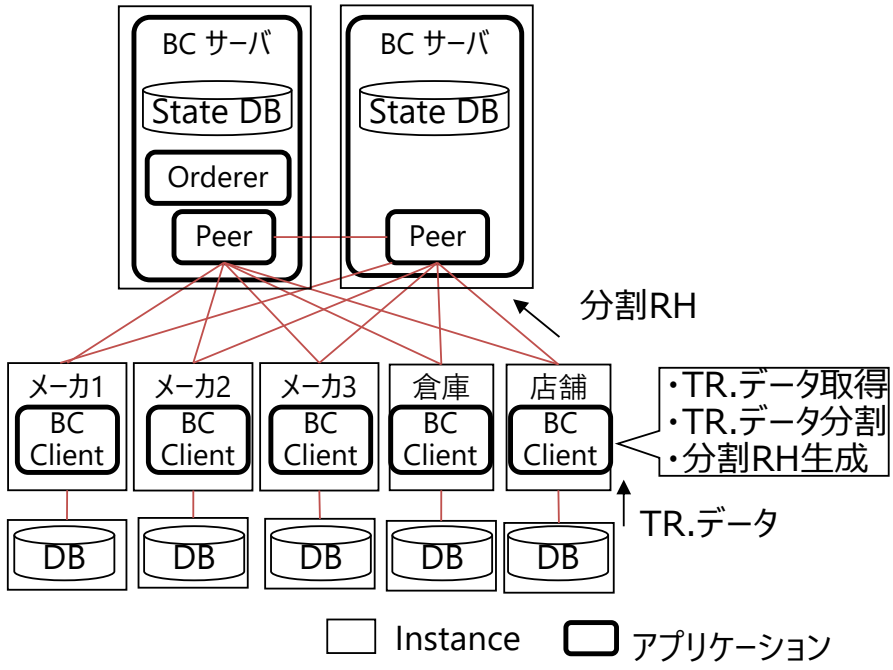

図 3-12 評価環境の構成

ば書き込みが行われ，or を設定すると，or で結ばれたうちの片方の peer の Endosement の結果が書き込みルールを満たせば書き込みが行われる[73].

DB，BC クライアント，BC サーバがそれぞれ異なる Ubuntu インスタンス上で動作する．表 3-9 に各構成要素の仕様をそれぞれ示す．ここで，表 3-9 の BC パラメータに示した各項目のいずれかを満たすと BC の Block の生成が行われる．

提案手法の場合, BC の 1Tx に複数の分割 RH を書き込むことで BC の Tx のデータサイズは増え Tx 数は少なくなる．Thakkar らによると，Tx 数が少ない場合は 1Block に書き込む Tx 数である Max Message Count（以降，M. M. Count と称す）を小さくしたほうが処理性能は向上する[76]．そこで，M. M. Count を最小値である 1Tx で測定する．

なお，評価環境の制約上 BC サーバ数は最大 2，BC クライアント数は最大 5 件に留まる．今回は仮に国内の年間 1,000 億のコンビニ商品をメーカ，倉庫，店舗までの TR.データを 5 つの BC クライアントで処理した場合を想定した評価となる．しかし現実には国内全体のコンビニ商品のトレーサビリティ管理を行う場合，特に BC クライアント数は 1 クライアントで複数拠点の TR.データの処理をまとめて行うことを想定しても数十から数百に及ぶと考えられる．このような BC サーバや BC クライアントの台数が増大した場合の評価は今後の課題とする．

表 3-10 に各拠点に対する入力パラメータ値を示す．TR.データ数は表 3-1 からメーカ全体と店舗で約 3,600Tx / sec，倉庫は約 192Tx / sec で十分であるが，各拠点 7,200 件

表 3-9 各構成要素の仕様

項目		仕様
DBMS		Elastic Search
BC Platform		Hyperledger Fabric V1.2, Couch DB
BC パラメータ	Max Message Count	1Tx
	Batch Time out	2.0 sec
	Preferred Max Bytes	1,024kB
クラウド環境	CPU	Intel Core Processor (Skylake, IBRS) 2GHz
	Num. of vCPU	12
	RAM	16GB
	Disk	42GB
	vOS	Ubuntu 20.04 LTS

表 3-10 入力パラメータ値

項目	倉庫	メーカおよび店舗
(a) TR.データ数（件）	7,200	7,200
(b) n_h：1 分割 RH におけるハッシュ数（件）	6	5
(c) n_c：メーカ数	3	1
(d) BC に書き込む合計分割 RH 数（件） ((a) * (c) / (b))	3,600	1,440

/ sec と設定した．これは次の理由による．まず表 3-8 を見ると，各拠点で最も分割 RH 数が多いのは店舗で，3,171 件 / sec の分割 RH の書き込みが生じる．この分割 RH の書き込み件数を模擬したいが，実験環境では倉庫以外では TR.データの分割を行う機能が実装できなかったことから, 倉庫でメーカ数を 3 として BC に書き込む分割 RH 数 3,600 件 / sec を店舗の代わりに倉庫で担保することにした．また，1 つの分割 RH に含まれる各イベントの TR.データは本来複数件だが実験環境では 1 件ずつとし，各イベントの TR.データ数も同じとしている．そのため実験環境では,

分割 RH 数 = メーカ数 * TR.データ数 / 各拠点のイベント数

という関係が成り立つ．倉庫のイベント数は n_h に等しく 6，メーカ数は 3 のため,

TR.データ数 = 3,600 * 6 / 3 = 7,200

となる．他のメーカおよび店舗でもこれに合わせて TR.データ数を 7,200 件として，BC への分割 RH の書き込みが多く発生する状況でも問題なく書き込み可能かを評価した．

3.5.4 書き込み性能の評価結果

表 3-11，表 3-12 にそれぞれ倉庫と他拠点（メーカおよび店舗）で n_r を変えたときの BC に書き込み可能な分割 RH 数を示す．表 3-11，表 3-12 の 1 列目の No が同じ行は，同じ回に行った測定結果である．ここで（f）BC の Tx のスループットは，5 分間（300 秒）書き込みを繰り返し，BC に書き込めた Tx 数を 300 秒で割った値である．

表 3-11 を見ると，倉庫で n_r = 1,800 とすることで（g）BC に書き込み可能な分割 RH 数は 3,515 件/sec に達しており，また表 3-12 においても n_r = 720 とすることで 1,404 件 / sec の分割 RH の書き込みが可能である．このことから現在達成できている倉庫および他拠点の TR.データの書き込み性能は，コンビニ電子タグ 1,000 億宣言の時代に十分耐えうる値である．ここで倉庫において n_r を 1,800 に設定することで，BC の 1Tx 当たりの書き込み件数およびサイズを増やして書き込み回数を減らしているが，文献[76]によ

ると BC の書き込み性能を担保する手段として一般的に行われており，現実的な手段と考える．なお，倉庫で n_r=900，店舗，メーカで n_r=360 の時は計測の途中から書き込み完了の応答の無くなるエラーが生じた．これは，BC の 1Block に書き込む Tx 数である Max Messsage Count を 1Tx としたのに対して，倉庫，店舗，メーカで入力 BC Tx 数を 4Tx / sec（表 3-11，表 3-12 の（e）列参照）としているため，BC の Block 生成が追い付かなくなったことが原因と考えられる．

今回の評価結果については，仮に全国のコンビニ商品の倉庫，店舗，メーカの商品を合計 5 つの BC クライアントで取り扱った場合の評価であり，実際には BC クライアントは増えると考える．今回の評価で，スループット低下が起きない BC サーバへの秒間の書き込み Tx 数は，各クライアントで多くて秒間 2Tx（表 3-11，表 3-12 の（e）列参照）であり，合計 5 クライアントの構成のため，全体で 2 * 5 = 10（Tx / sec）以内である．そのため，BC クライアントが増えた場合には，BC サーバへの合計の BC の秒間書き込み Tx 数を 10Tx / sec 以内に抑えることが必要と考える．そのために，複数の拠点でまとめて BC クライアントを設置することや，各クライアントの BC サーバに対する書き込み頻度を適切に下げることが求められる. 今後そのようなクライアントが増えた

表 3-11 倉庫における評価結果（分割 RH 数：3,600 件 / sec）

No.	n_r	(e) 入力 BC Tx 数 (Tx / sec)	(f) N_t : BC の Tx のスループット (Tx / sec)	(g) N_{RH} : BC に書き込み可能な分割 RH 数 (Tx / sec) $N_{RH} = n_r * N_t$
1	900	4	error	error
2	1,800	2	1.95	3,515

表 3-12 メーカおよび店舗における評価結果（分割 RH 数：1,440 件 / sec）

No.	n_r	(e) 入力 BC Tx 数 (Tx / sec)	(f) N_t : BC の Tx のスループット (Tx / sec)	(g) N_{RH} : BC に書き込み可能な分割 RH 数 (Tx / sec) $N_{RH} = n_r * N_t$
1	360	4	error	error
2	720	2	1.95	1,404

場合のシステム構成や各拠点からの BC への書き込み時のデータのまとめ方，タイミング等について詳細化する．

なお，3.4 節で述べた（a）TR.データ，（b）ハッシュ，（c）RH，（d）分割 RH の各書き込み方法の処理性能を比較した場合，BC への書き込み量が少ないほど書き込み性能は高くなると考えられることから，書き込み性能が高い順に，「（c）RH＞（d）分割 RH ＞（a）TR.データ ＞（d）ハッシュ 」となると考える．（a）と（b）の処理速度はほぼ同等だが，（b）は（a）よりもハッシュを生成するための処理時間が掛かる．

3.6　検索・真正性検証処理の性能評価

本節では，3.3.5 節で説明した検索・検証処理の処理性能の評価を述べる．

3.6.1　評価内容と方法

評価内容について説明する．3.3.5 節の図 3-11 の SC の構成において，店舗で一般消費者が SSCC：0001-1 の商品を購入前に，この商品の製造からの TR.データのトレースバックを行う場合を想定する．各拠点の DB には，3,600 件の TR.データが入った状態での検索と真正性検証処理を行う．

評価方法を説明する．表 3-9 に示したクラウド環境上で，TR.データの検索・真正性検証処理に必要な 6 つの処理を実行した場合の処理時間を評価した．この中で処理 3，5 の処理時間については机上での見積もり，他の処理時間は実測である．

処理 1.　検索クエリの商品 ID をもとに，出荷データテーブルから各拠点の出荷ファイル名，BC の Key を取得
処理 2.　BC から各拠点の TR.データ名と分割 RH を取得
処理 3.　処理 1，2 の結果を一時格納テーブルに格納
処理 4.　各拠点の DB から TR.データを取得
処理 5.　検索対象の商品に絞り込む処理（図 3-10 の S7 に該当）
処理 6.　TR.データから分割 RH を生成し，BC から取得した分割 RH と比較

3.6.2　評価結果

表 3-13 の 2 列目に 3,600 件のデータの場合の検索・検証処理時間の評価結果を示す．約 1.2 秒でトレースバックにおける検索・検証処理を完了できる見通しである．

表 3-13 データ検索と検証の処理時間（sec）

	3,600 件	1,000 億件
処理 1	0.1	2.0
処理 2	0.1	0.5 ～ 1.0
処理 3	0.2	0.2
処理 4	0.2	1.0 ～ 2.0
処理 5	0.4	0.4
処理 6	0.2	0.2
合計	1.2	4.3 ～ 5.8

次に，本サービスが実用化され，仮に 1,000 億件のデータが各拠点にある場合の検索・検証処理時間を見積もる．処理 1 については，出荷テーブルのサイズが大きくなることで検索時間も増大する．そこで商品の分類（食品，日用品，書籍等）や商品の製造地域などで出荷データテーブルを分割して検索空間の広がりを制限すれば，出荷データテーブルはハッシュテーブルを用いていることから，検索時間を抑えられる．例えば文献[77]では 1,000 万件（10^7 件）のハッシュテーブルからのデータ検索に掛かる時間は 1.01 秒と述べている．そこで，1,000 億件（10^{11} 件）の出荷データテーブルのデータを 10,000（10^4）分割すれば，検索対象のデータが分割されたどの出荷データテーブルに属するかを検索するのに要する時間を合わせても 2 秒以内に検索ができるものと考える．今後出荷データテーブルの分割方法について，詳細化する必要がある．

処理 2 についても同様に Hyperledger Fabric のチャネルの機能を用いてデータを商品分類や製造地域で分けることで検索時間を抑えることができる．処理 4 については DB の検索速度と NW の通信速度が課題となるが，例えば DB を拠点ごとに配置せずに，企業ごとに持つなどして集約させるとともに，分散 DB を用いて商品分類ごとに分散して保持するなどすれば 1.0～2.0 秒の取得時間にできると考える．

処理 3，5，6 は総データ件数が変わっても処理するデータ量は変わらないため，処理時間は 3,600 件の場合と同等である．以上を踏まえると，1,000 億件のデータがある場合でも合計で約 4.3～5.8 秒の処理時間と考えられ，実サービスにおいて検索者が大きなストレスなく検索を行うことが可能と考える．

3.7　まとめ

本章では，分割 RH 方式による BC 利用トレービリティ管理システムを提案した．本システムでは，商品の TR.データのハッシュ値を書き込む際に，製造元ごとに分けて書き込むことでデータの秘匿性を保証し, かつ, 同一商品の TR.データをまとめた分割 RH を BC の 1Tx に複数書き込むことで高速性を担保する．BC の書き込み性能と秘匿性の評価を行い，コンビニ電子タグ 1,000 億宣言実現時に想定される年間 1,000 億個の商品を取り扱い可能であることを示した. またサービス提供者と閲覧者に対するデータの秘匿性の要件を満足することを確認した.

主な今後の課題として次の 8 点が有る．1 点目は BC では保証できない BC に書き込む前の入力データの不正を防ぐ方法の検討である．例えば 3.3.2 節で述べた，複数メーカの商品情報が含まれた TR.データを分割する際に改ざんの余地が有ることへの対策が必要である．2 点目は 3.3.2 節で述べた長期に出荷・販売されない商品に対する TR.データの分割 RH 生成タイミングと BC 書き込みタイミングの検討が必要である．3 点目の課題としては，同じく 3.3.2 節の最後に述べた，分割 RH 生成時の同一拠点内の TR.データを検索する処理時間の評価が必要である．4 点目の課題としては，3.4 節の表 3-8 で述べたように，店舗においては秒間 3,171 件の分割 RH 書き込みが生じる．5.4 節の性能評価では処理可能との結果を得たが，BC データ維持コストの削減の観点で，例えば店舗でも同じ梱包箱に梱包されていた商品が全て販売されたら，それら商品で 1 回分割 RH を生成するなどして分割 RH の書き込み件数を減らすことが必要である．5 点目は，3.6.1 節で述べたように出荷データテーブルで扱うデータ件数が増大した場合の分割方法の検討が必要である．6 点目は，実際に国内の各店舗で販売されるコンビニ商品のトレーサビリティを取り扱うためのシステム構成や，BC の Peer のチャネル構成，コンセンサスアルゴリズム，各拠点からの BC への書き込み時のデータのまとめ方，タイミング等について詳細化する必要がある．7 点目には，提案手法をコンビニ商品以外の SC に適用した場合に, 特に製造業の組み立て工程では異なるメーカの部品が複数回組み合わさることで分割 RH 数が増大する課題がある．そのような分割 RH 数の増大への対策が必要である．最後に 8 点目として，3.3.2 節で述べた分割 RH 生成時に，TR.データ中の他メーカの商品の情報を伏せ字以外にする場合の処理速度等の評価である．例えば，他メーカの商品名を伏せ字にする以外の方法として，最初から TR.データをメーカごとに分けて生成する方法や他メーカの商品情報を伏せ字にせず他メーカの商品名を削除する方法が考えられる.

第4章
TEE を用いた関数型暗号による秘匿配送マッチングシステムの処理時間の評価

4.1 緒言

物流業界では，荷主がトラックを直接手配できない場合などに，サービス提供者がトラックを手配する配送マッチングサービス（以下，マッチングサービスと称す）が行われている[4]. 一般的なマッチングサービスでは，マッチング候補となるトラックのルートの情報を予め収集しておき，荷主から届いた追加オーダの配送元・配送先の情報とルートの情報をもとに，適切なトラックにその追加オーダを割り当てる．ここで，荷主の追加オーダとトラックのルートには，出発地・到着地の所在地などの機微情報が含まれるので, サービス提供者を含めた他者（追加オーダは荷主以外, ルートはトラック以外）に対して追加オーダとルートを秘匿することが望まれる．

そこで, 秘密計算を用いて追加オーダとルートの中身を秘匿した状態でマッチングを行う秘匿配送マッチングに取り組む. 国内大手のマッチングサービス業者と同等のマッチング件数を実現しようとした場合，4.2.5 節で述べるように 289 件のトラックのルートが含まれる 1 回のマッチングを 3.6 秒で処理する必要がある．秘密計算の手法の例として，暗号文のままで線形演算と乗算が可能な準同型暗号[17]があるが，同手法ではこの制約を満たすのは困難である．

一方で秘密計算を高速に実現する方法として, TEE によって秘密計算の一手法である関数型暗号を実現する方法が知られている[20]．関数型暗号は秘密鍵の利用者が秘密鍵で暗号文を復号する際に, 任意の演算結果を得ることができる暗号である[19]. また TEE は外部にデータを秘匿することが可能な CPU の保護領域であり，高速な処理が可能である．しかし，TEE による関数型暗号においても，TEE 内でルートと追加オーダを平文に復号する処理などのオーバヘッドが懸念される．そこで本章では，秘匿配送マッチングに TEE による関数型暗号を適用した場合の処理時間を明らかにするとともに，サービス提供者を含めた他者への秘匿性が満たせるか検証することを目的とする．

以降，4.2 節で秘匿配送マッチングの課題，4.3 節で TEE による関数型暗号の秘匿配送マッチングの処理内容，4.4 節で評価環境の実装と評価，4.5 節でまとめを述べる．

4.2　秘匿配送マッチングの課題

本節では，配送マッチング問題および秘匿配送マッチングの処理概要を述べ，課題を説明する．

4.2.1　配送マッチング問題

マッチングサービスにおいて，複数のトラックのルートと 1 つの追加オーダに対して，その追加オーダをどのルートに割り当てるかを求める問題を配送マッチング問題と定義する．マッチングサービスでは，サービス提供者がルートと追加オーダをもとに配送マッチング問題をマッチング Platform（以降，マッチング PF と称す）上で解いて，追加オーダの割り当て先のトラックを決める．ここでルートは，マッチングを行う前にトラックの所属する運送会社等が事前に決定しており，そこに荷主からの追加の配送依頼（追加オーダ）を割り当てる．トラックは 1 回の配送業務で最初の出発地（起点と称す）から最終到着地（終点と称す）まで，途中の各地点で荷物の荷積み・荷降ろしを行いながら向かう．ルートはトラックの配送業務ごとに作成され，トラックが立ち寄る複数の地点が順番に記載される．例えば「起点→地点 1→地点 2→終点」と記載する．また各地点には緯度・経度が紐づいており，ルートは地点を節点，地点間を結ぶ直線を辺とするグラフとみなすことができる．追加オーダには，荷主がトラックに運んで欲しい荷物の出発地（発送元）と到着地（配送先）が記載されている．

4.2.2　割り当て先のトラックの決定方法

マッチングサービスのサービス提供者は以下の条件を加味して，追加オーダの割り当て先のトラックを決定する．なお，荷物の積載量や各地点への到着時間の制約も加味することも可能だが，本章では対象外とする．

条件 1. トラックは追加オーダ中の出発地から到着地に直接向かい，途中でトラックが立ち寄る予定のルートに含まれる他の地点には立ち寄らない

条件 2. 追加オーダを割り当てた場合の追加距離が最小となるルートの辺に，追加オーダを割り当てる

条件 1 について，図 4-1 を用いて説明する．図 4-1 は 2 つのトラックが存在する場合に，地点 7 から 8 へ配送する追加オーダを割り当てる配送マッチング問題の例を示している．条件 1 はトラックが地点 7 から 8 に荷物を運ぶ途中に，そのトラックが立ち寄

る予定のルートに含まれる他の地点（例．トラック 1 であれば地点 1，2，3）には立ち寄らないことを意味する．これは，途中で他の地点に寄って別の荷物の荷積み・荷降ろしがあると，追加オーダの出発地（地点 7）で積んだ荷物がトラックの荷台の奥に行ってしまい，到着地（地点 8）における荷降ろしがしづらくなる，また別の地点での荷降ろし時に，出発地（地点 7）で積んだ荷物が荷降ろし作業を阻害する，といった懸念が有るためである．

次に条件 2 について説明する．一般的な配送計画問題では，トラックドライバの労働時間やトラックの燃料代の負担が過多とならないように，走行距離が最小となるようにルートを策定する[78]．配送マッチング問題においても，マッチングで割り当てた追加オーダが上述の負担とならないよう，追加距離が最小となるルートの辺に追加オーダを割り当てる．なお追加距離は，地図上の経路に沿った距離で評価することも考えられるが，経路に沿った距離がすぐに得られない場合には 2 地点$((x_1, y_1), (x_2, y_2))$間の直線（ユークリッド）距離$(\mathrm{sqrt}\{(x_2-x_1)^2+(y_2-y_1)^2\})$，または碁盤目状に道路が有る場合にはマンハッタン距離（$|x_2-x_1|+|y_2-y_1|$）で評価することが一般的であり，文献[78]でも直線距離を用いている．そこで本研究でも追加距離は直線距離で評価するが，マンハッタン距離に変更することも容易である．ここでルートの地点 i から地点 j の辺の途中に出発地 k から到着地 l の追加オーダを割り当てた場合の追加距離 L は，

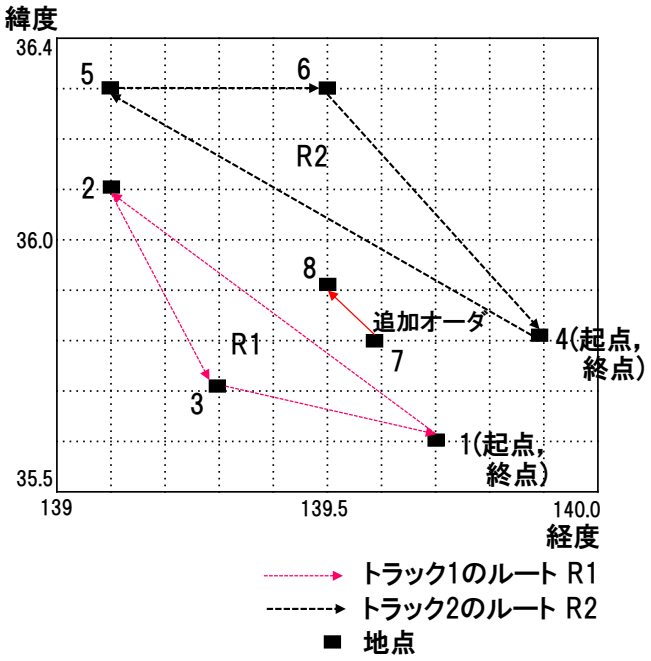

図 4-1 配送マッチング問題の例

L = 地点 i と出発地 k の直線（ユークリッド）距離 +
出発地 k と到着地 l の直線距離 + 到着地 l と地点 j の直線距離 -
地点 i と地点 j の直線距離　　（４－１）
と定義される.

4.2.3　配送マッチングで用いるデータと処理内容

配送マッチング問題で用いるトラックのルートの内容を表 4-1，荷主の追加オーダの内容を表 4-2 にそれぞれ示す．ルート，追加オーダとも，1 回の配送（辺）の出発地・到着地は，予め各トラックおよび荷主の端末で保持している住所情報のテーブルに登録されている場所 ID を用いて指定される．またトラックのルートには，複数の辺（1 回の配送に該当）が存在する. 次に, 各トラックと荷主が用いる住所情報を表 4-3 に示す. 住所情報は緯度・経度と住所（地番）が登録されており，各トラックおよび荷主ごとに自身の端末で保持し，マッチング PF や他者には共有されない．配送マッチングでは，ルートと追加オーダ中の出発地・到着地の地点 ID を緯度・経度に変換してマッチング PF に送信し，マッチング PF 上で割り当て先のトラックを決定する.

表 4-1 ルートの内容

ルート ID	辺 ID	出発地	到着地
1	1	1	2
1	2	2	3
1	3	3	1

表 4-2 追加オーダの内容

オーダ ID	出発地	到着地
1	7	8

表 4-3 住所情報

地点 ID	緯度(°)	経度(°)	住所
1	35.6	139.7	横浜市 xx 区 xx 町 292
2	36.1	139.1	相模原市 xx 区 xx 町 12
…	…	…	…

マッチング PF は各トラックのルートを 1 日の配送が始まる前に予め入手しており，トラックが起点を出発する前にマッチングが行われ，追加オーダがマッチング PF に届くとマッチング処理が始まる．マッチング PF は各トラックのルートのどの辺に割り当てたら追加距離最小となるかを探索し，追加オーダの割り当て先を決める．例えば図 4-1 の場合，2 つのルート R1，R2 にはそれぞれ 3 つの辺（例．ルート R1 であれば地点 1→2，地点 2→3，地点 3→1）が存在する．このそれぞれの辺に対し，追加オーダを割り当てた際の追加距離が最小となる辺に追加オーダを割り当てる．なお，トラックや荷主がマッチング結果を承諾しなかった場合は，そのトラック以外のトラックのルートで最も追加距離が短いルートの辺を割り当て先とする．ここでマッチングに用いるトラックのルートは日々変わることを想定しており，トラックはマッチング PF にルートを毎日決められた時刻までに送り，マッチング実施時にはマッチング PF に揃っているルートを用いてマッチングを行う．またトラックの事情でルートに変更が有った際は，変更後のルートをマッチング PF に送付してルートを更新する想定である．

4.2.4　秘匿配送マッチング

荷主とトラックにとって，サービス提供者を含めた他者に出発地・到着地の緯度・経度が分かると，Web 等で検索すれば取引先企業名が分かり望ましくない．そこで，トラックと荷主が，出発地・到着地の緯度・経度をサービス提供者も含む他者に秘匿した状態でマッチングを行う秘匿配送マッチングに取り組む．図 4-2 に秘匿配送マッチングの概要を示す．以降，秘匿配送マッチングを実施するマッチング PF を秘匿配送マッチング PF と称す．なおトラックと荷主のクライアント端末，および秘匿配送マッチング PF とクライアント端末間の通信路のセキュリティについては既存の方法で担保するものとし，本章の処理時間および秘匿性の評価対象外とする．

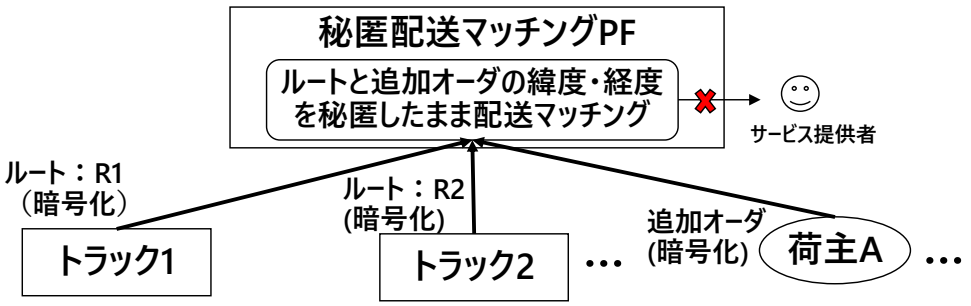

図 4-2 秘匿配送マッチングの概要

4.2.5　処理時間の目標

配送マッチングの国内大手のトランコム社（以降, T 社と称す）では日々6,000 件（2021 年度）のマッチングを 45 拠点で成立させている[79]．T 社では，秘匿配送マッチングのように追加オーダとルートを秘匿した状態でのマッチングは行っていない．本章では，追加オーダとルートを秘匿した状態で T 社と同等のマッチング件数をめざす．ただし処理時間の評価対象は，PF がルートと追加オーダを受信して割り当て先のトラックを導出するまでの時間とする．ここで，マッチングサービスは荷主が翌日のトラックを手配できない場合に利用されることが多く，前日の午後（13-17 時）に利用が集中すると考えられる．仮に 2 / 3 の 4,000 件がこの 4 時間に集中した場合，1 時間あたり 1,000 件（1 回あたり 3.6 秒以内）のマッチングを成立させる必要がある．

次に 1 回のマッチングで取り扱うルート数について説明する. T 社では 1 日 13,000 件のトラックの空車情報を扱っており[1]，空車情報 1 件に各トラックのルート 1 件が含まれる．そこで，T 社と同様に 45 拠点で分けてマッチングを行うものとすると，1 回のマッチングでは，13,000 / 45 = 289 件のルートを扱うことになる．そこで秘匿配送マッチングでは，T 社と同規模のサービスを一台のサーバで実現するものとし，289 件のルートが含まれるマッチングを 1 回あたり 3.6 秒以内に成立することを目標とする．

ここで各ルートに含まれる辺の数について説明する. 宅配の近距離便であれば各住宅に配送することから，一回の配送業務で数十地点に立ち寄ると考えられるが，T 社では宅配の近距離便のマッチングは扱っておらず，BtoB の配送がマッチング対象である．BtoB の配送の場合，各地点における荷積み・荷降ろし作業に時間が掛かることや，立ち寄り先が店舗や工場などであり, 各地点間の距離も宅配の近距離便と比べて離れていることから，1 回の配送業務で立ち寄る地点数は始点，終点を除いて平均 3 か所（例：始点を朝出発して 3 地点で荷降ろしを行い, 夕方始点に戻る. ）と考えられる. そこで，各ルートには 4 つの辺が含まれるものとする．

4.2.6　秘匿配送マッチング実現の課題

秘匿配送マッチングにおいては，秘匿配送マッチング PF 上で暗号文のルートと追加オーダから，追加オーダをルートの各辺に割り当てた際の追加距離を計算し，その最小値を取得する．データを暗号化した状態で計算する技術は，一般に秘密計算と呼ばれ，

1 https://www.trancom.co.jp/transport/ (23/7/4 閲覧)

秘密計算の手法の例として，暗号文のままで線形演算と乗算が可能な準同型暗号がある．しかし処理時間の課題が有ることが知られている[80]．例えば準同型暗号のライブラリとして，Microsoft の SEAL を用いて表 4-4 に示すクラウド環境上で秘匿配送マッチングの処理を実装した場合, 4 つの辺が含まれる 100 件のルートに対する処理時間を計測したところ，1 回のマッチング（追加距離の算出と割り当て先の辺の決定）に約 42 秒掛かることが分かった．ここで SEAL では，準同型暗号のベースとなる格子暗号の方式として，実数を整数に見立てて計算する CKKS（Cheon, Kim, Kim, Song）方式を用いた．そのため，SEAL で 289 件のルートが含まれるマッチングを 3.6 秒以内に成立することは実現困難であり, 他の準同型暗号のライブラリでも SEAL と処理速度は同等であることから準同型暗号による目標達成は困難である[80]．また，準同型暗号よりも高速な秘密計算の手法として秘密分散[84]があるが，追加距離計算における乗算の処理などにおいてサーバ間の通信のオーバヘッドが発生し, やはり処理速度が課題になると考えられる．

そのため，ハードウェア的に高速な秘密計算が実施可能であり，近年注目を集めている TEE を用いて秘匿配送マッチングシステムを構築し，処理時間およびデータの秘匿性を検証する．ここで処理時間については，TEE を用いた場合には TEE の内部でルートと追加オーダを平文に復号してからマッチング処理を行うが, その復号に掛かる処理時間などのオーバヘッドが懸念されるため，4.2.5 節で述べた処理時間の制約を満たせるかは不明という課題がある．

表 4-4 準同型暗号による実装環境

項目		仕様
準同型暗号ライブラリ		Microsoft SEAL v3.5.6 CKKS 方式
言語		C++
クラウド環境	vOS	Ubuntu16.04LTS
	CPU	Intel Core Processor (Skylake, IBRS) 2GHz
	RAM	16GB
	Disk	42GB

4.3　TEE を用いた関数型暗号の適用

本節では, TEE による関数型暗号を用いた秘匿配送マッチングの処理内容について説明する.

4.3.1　TEE による関数型暗号

Fisch らは TEE として SGX を用いてハードウェア的に関数型暗号を実現することで, TEE の外に入力値を秘匿した状態で任意の関数を高速に処理し, その結果を取得できる IRON と呼ばれるシステムを開発している[20]．そこで本章では，IRON と同様に TEE として SGX を用いて関数型暗号を実現し, 秘匿配送マッチングに適用する. ここで SGX では，主記憶装置上に形成された Enclave と呼ばれる保護領域内で，秘密情報を保持しながらプログラムを実行できる．SGX の概要は 4.3.2 節で説明する．また IRON では，鍵生成者である Key Manager が SGX にルートと追加オーダの復号用秘密鍵を渡す際に，Remote Attestation と呼ばれる Enclave で実行されるバイナリが，実行者の意図したものであるか外部から確認することができる機能を用いる．このように Key Manager は単に鍵を生成するのみならず，Enclave 内の処理の妥当性を確認する役割を担う．図 4-3 に IRON[20]のシステム構成を示す.

IRON では関数型暗号の処理を行う PF である Decryption Node PF において，関数型暗号の関数である関数 f（図中では関数 F）の演算を行う Function Enclave（以降，FE と称す）と，Key Manager から復号用秘密鍵 sk_{pke} を受け取り，FE を認証した後に復号用秘密鍵 sk_{pke} を提供する Decryption Enclave（以降，DE と称す）の 2 つの Enclave が存在する. DE が FE を認証する際は, 同一 CPU 上の Enclave 間の認証である Local Attestation が用いられる．Local Attestation，Remote Attestation ともに，Enclave を認証する際には Enclave プログラムの生成（ビルド）時に Enclave プログラムから計算するハッシュ値 mrenclave (Measurement Registor of Enclave) が用いられる．mrenclave は Enclave プログラムに依存する．図 4-3 中の主な略号の意味は次の通りである.

・(pk_{pke}, sk_{pke})：関数型暗号の入力データの暗号用公開鍵，復号用秘密鍵

・関数 f : FE で行う関数型暗号の演算を表す関数

・(vk_{sign}, sk_{sign})：関数 f の署名検証鍵，署名用秘密鍵

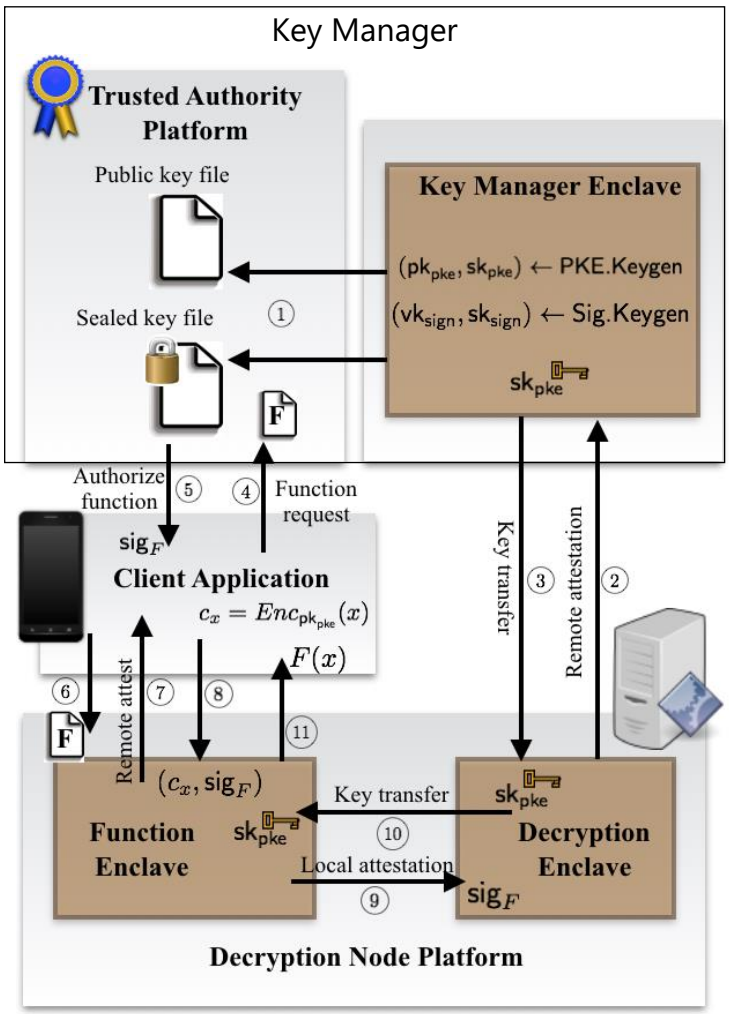

図 4-3 IRON のシステム構成（文献[20]の figure2 を引用）

また，(pk_{pke}, sk_{pke}) と (vk_{sign}, sk_{sign}) の鍵を管理する Key Manager の構成要素として，鍵生成を行う Key Manager Enclave と，FE の中で行われる関数 f の処理に改ざんがないことを認証する役割を担う Trusted Authority Platform が存在する．(vk_{sign}, sk_{sign}) の用途，および関数 F の認証処理の手順は後述の 4.3.2.2 節で説明する．

次に関数型暗号では，秘密鍵の利用者が秘密鍵で暗号文を復号する際に，任意の演算結果を得ることができるものであり，複数の暗号文を入力として秘密鍵を施すと演算結果が得られる多入力関数型暗号も存在する[52]．秘匿配送マッチングの処理は，複数のルートと 1 つの追加オーダの暗号文から追加距離最小となる割り当て先のルートの辺を求める多入力関数型暗号に該当し，サービス提供者は演算結果である割り当て先のルートの辺を FE から取得する．

4.3.2　SGX の概要

本節では SGX[22]の概要について説明する．SGX では Enclave と呼ばれる保護領域を保持している TEE であり，秘密情報を暗号化して Enclave に入力し，Enclave 内で復号して必要な演算を実行後に結果を Enclave 外に出力することで, 秘密計算が可能である．次に Remote Attestation，Local Attestation で用いられる mrenclave については，Enclave をビルドした際に生成されるイメージファイル自体に固有なハッシュ値であり，Enclave 中に記述したプログラム（ソースコード）のハッシュ値と，Enclave のイメージファイル生成に必要な設定ファイル, マシンのファームウェア情報等から生成されるハッシュ値をバイナリ的に結合したものである．

次に Remote Attestation については，Enclave がリモートマシン（ユーザとは別のマシン）に構築されている場合，Enclave の処理がユーザの意図しているものと一致している事をユーザが Intel の IAS (Intel Attestation Service) と呼ばれる SGX の認証サーバにも問い合わせて認証する処理である[22]．IAS のサーバでは，Enclave から取得した CPU 製造番号および mrenclave の中で Enclave のソースコードに依存しない部分が IAS のサーバで予め保持している情報と整合するか確認され，Enclave が認証される．

4.3.3　秘匿配送マッチングシステムの全体像

図 4-4 を用いて，TEE による関数型暗号秘匿配送マッチングシステムの全体像を説明する．本システムの構成は IRON と基本的に同一であるが，図 4-3 の IRON の Client Application が秘匿配送マッチング PF の PF. App に置き換わり，PF. App が FE の中で秘匿配送マッチングの処理を行う．IRON では任意の関数型暗号の処理（図中の関数 f）を扱えるように設計されているが，本章では関数 f は秘匿配送マッチングに特化して追加距離計算と割り当て先トラック決定を行うように実装している．図 4-4 で，秘匿配送マッチング向けに実装した箇所を点線で示す．図 4-4 における各略号の意味は以下の通りである．

・DE：Decryption Enclave の略

・FE：Function Enclave の略

・KME：Key Manager Enclave の略

・PF App.：PF の Enclave 外のアプリ

・Key Manager App.：Key Manager の Enclave 外のアプリ

・(pk_{pke}, sk_{pke}) ：ルート・追加オーダの暗号用公開鍵, 復号用秘密鍵. pk_{pke} は Key Manager

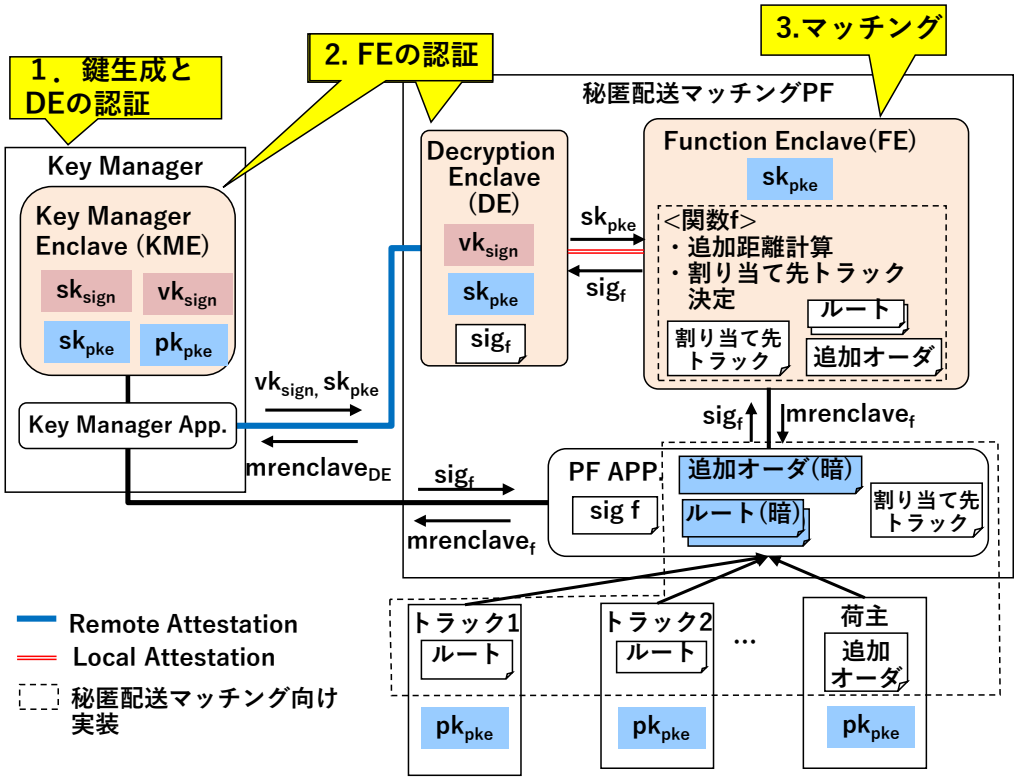

図 4-4 秘匿配送マッチングシステムの全体像

が KME で作成してサービス提供者の PF.App に送り，その後トラックと荷主に配布．sk_{pke} は Key Manager が KME で作成後，Remote Attestation で Key Manager から DE に送付．

・関数 f：FE で行う処理の関数

・(vk_{sign}, sk_{sign})：関数 f の署名検証鍵，署名用秘密鍵．Vk_{sign} は Key Manager が KME で作成後，Remote Attestation で Key Manager から DE に送付．sk_{sign} は Key Manager が KME 作成後，KME で保持．

・$mrenclave_f$：FE から生成されるハッシュ値である mrenclave 値

・$mrenclave_{DE}$：DE から生成されるハッシュ値である mrenclave 値

・sig_f：$mrenclave_f$に対する電子署名

関数型暗号による秘匿配送マッチングシステムは，1．鍵生成と DE の認証，2. FE の認証，3．マッチング処理の三つの処理からなる．これらの処理を構成する要素として秘匿配送マッチング PF と Key Manager が定義される．秘匿配送マッチング PF には DE と FE の 2 つの Enclave，および PF App が内部モジュールとして存在する．また，Key Manager には内部モジュールとして KME と Key Manager App.が存在する．Key Manager

は，IRON と同様に Trusted Authority の役割を兼ねており，秘匿配送マッチング PF を運営するサービス提供者の不正を監視する．KeyManager は，サービス提供者とは独立した立場の第三者認証機関が担当する想定である．

ここで，DE の認証は Key Manager が Remote Attestation（以降，R.A.と称す）を用いて行い，通常の R.A.に加えて，Key Manager が DE の mrenclave 値である $\text{mrenclave}_{\text{DE}}$ の正解値を予め保持しておき，R.A.で DE から送付される $\text{mrenclave}_{\text{DE}}$ が毎回同じであるかを確認することで，Key Manager が DE 内のソースコードの改変が無いか確認する．また FE の認証は，DE が FE の中の処理に改ざんがないことを認証する処理である．詳細は 4.3.3.2 節に示す. ここで, FE と DE を分離した理由を説明する. KME では $\text{mrenclave}_{\text{DE}}$ の値（正解値）を保持しているが，DE と FE では FE の処理の書き換え頻度が高いことから，もし FE と DE を分離しなかった場合，KME の正解値の書き換えが頻繁に起きてしまい管理上望ましくない．そこで FE と DE を分離すれば，KME で保持する正解値の mrenclave は書き換え頻度の少ない DE の mrenclave で良く, FE と DE を分離しない場合と比べて，正解値の mrenclave の書き換え頻度を減らすことができる．

以降，秘匿配送マッチングシステムの各処理について説明する．

4.3.3.1 鍵生成と DE の認証

Key Manager は $(\text{pk}_{\text{pke}}, \text{sk}_{\text{pke}})$，$(\text{vk}_{\text{sign}}, \text{sk}_{\text{sign}})$ を KME 内で生成する．また，Key Manager App.は DE からの R.A.を受けて，DE を検証後に署名検証鍵 vk_{sign} とルート・追加オーダの復号用秘密鍵 sk_{pke} を提供する．その際，KME にて DE の mrenclave 値 $(\text{mrenclave}_{\text{DE}})$ が，KME は予め保持している正解値の $\text{mrenclave}_{\text{DE}}$ の値と比較して一致検証を行う．これにより，第三者による秘匿配送マッチング PF の成りすまし，サービス提供者や第三者の DE の改ざんを抑止する．ここで R.A.には Enclave の認証の際に，Intel 社の IAS に都度問い合わせる方式である EPID (Enhanced Privacy ID) 方式[81]を用いる

4.3.3.2 FE の認証

Key Manager App.は PF App.から FE の処理内容(関数 f)のハッシュ値である $\text{mrenclave}_{\text{f}}$ を受けて KME に送り，KME にて $\text{mrenclave}_{\text{f}}$ に sk_{sign} で署名を行い，その結果である sig_{f} を PF App.に送る．PF App.は sig_{f} を取得後 FE に送信する．

次に，FE は DE に対して Local Attestation を要求し，sig_{f} を DE に提供する．DE は署名検証鍵 vk_{sign} で sig_{f} を復号して得られる $\text{mlenclave}_{\text{f}}$ の値が Local Attestation（以降，L.A.と称す）で得られる $\text{mlenclave}_{\text{f}}$ の値と一致するか検証し，一致すれば FE に復号用秘密

鍵 sk_{pke} を提供する．これにより，FE の処理内容が Key Manager が署名した処理から改ざんがされていないことを担保する．

4.3.3.3 マッチング処理

トラックと荷主は PF App.から公開鍵 pk_{pke} を取得し，それぞれルート・追加オーダを暗号化して PF App.に送る．PF App.は FE に暗号化されたルート・追加オーダを送り，FE は sk_{pke} にてルートと追加オーダを平文に戻し，マッチング処理を行う．FE はマッチング処理の結果を PF App.に通知する．

4.3.4 鍵生成と DE の認証の処理詳細

まず，Key Manager が DE を認証して sk_{pke} と vk_{sign} を送付するまでの手順について，図 4-5 を用いて説明する．

(1) KME は鍵生成を行う

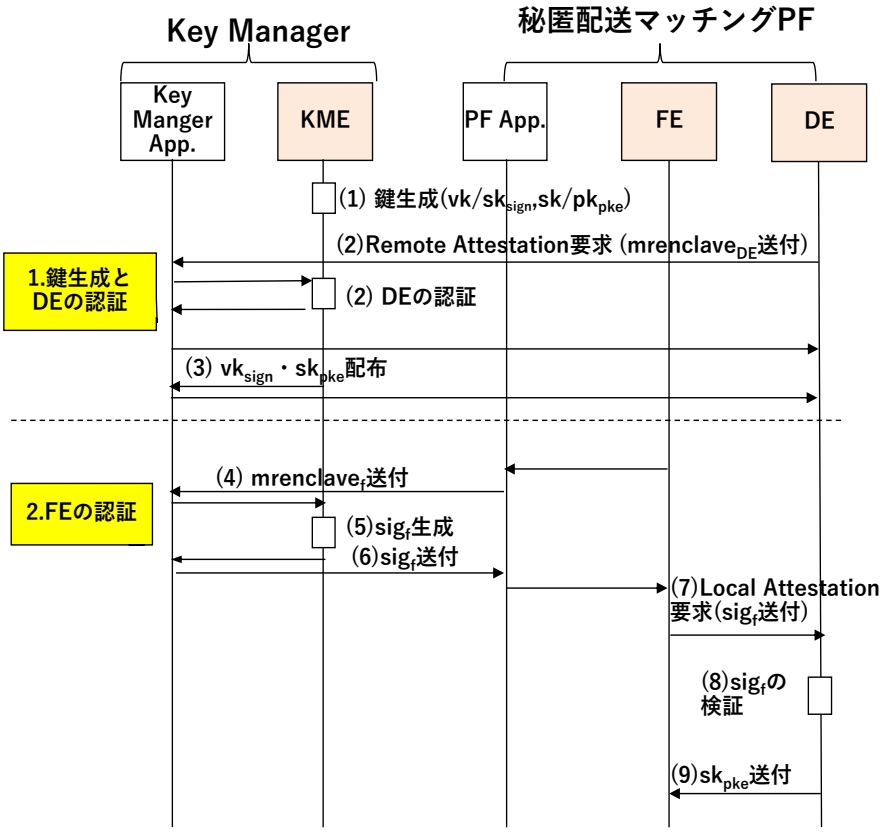

図 4-5 鍵生成と DE，FE の認証の処理フロー

(2) DE は R.A.を Key Manager App に要求

(3) Key Manager App.は sk_{pke} と vk_{sign} を DE に配布

sk_{pke} と vk_{sign} は DE が一度受け取ったら，秘匿配送マッチング PF のメモリに Sealing（SGX 内のデータを暗号化してメモリに書き込む処理）で保持しておき，PF のメモリから取得するが，DE の認証は定期的（例. 1 日に一回）に行い DE の不正を抑止する．

また DE の処理の変更が必要な場合は，Key Manager は DE から新たな $mrenclave_{DE}$ を取得し KME で保持している正解値の DE の mrenclave を置き替え，sk_{pke} と vk_{sign} を再発行する．

4.3.5 FE の認証

次に，DE が FE を認証して sk_{pke} を送付するまでの処理を同じく図 4-5 を用いて説明する．

(4) PF App.は $mrenclave_f$ を取得し，$mrenclave_f$ を Key Manager App.に送付

(5) KME は $mrenclave_f$ を sk_{sign} で署名して sigf を生成

(6) Key Manager App.は sig_f を PF App.に送付し，PF App は sig_f を FE に送付

(7) FE は DE に L.A.を要求し，sig_f を送付

(8) DE は sig_f を vk_{sign} で復号し，L.A.で取得した FE の mrenclave 値と一致するか検証

(9) DE は(8)の検証結果が True であれば，L.A.成立．DE は FE に sk_{pke} に送付

（4）～（6）は初回および FE の処理内容の変更や FE の追加があった場合のみ行い，sig_f は PF App.が一度 Key Manager から取得したら PF App.が保持する．一方（7）～（9）は定期的（例. 1 日に 1 回）に行うことで，Key Manager が署名を行った処理のみが秘匿配送マッチング PF 上で行われるようにする．

4.3.6 マッチング処理の詳細

図 4-6 を用いてマッチングの処理の詳細を説明する．なお(17)以降の処理は将来的に秘匿配送マッチングサービス事業を実現する際には必要となるが，本章における処理時間の評価対象には含めていない．

(10) PF App.は予め Key Manager から取得した暗号用公開鍵 pk_{pke} をクライアントに配布（初回のみ）

(11)トラックはルートを暗号化し，PF App.に送付

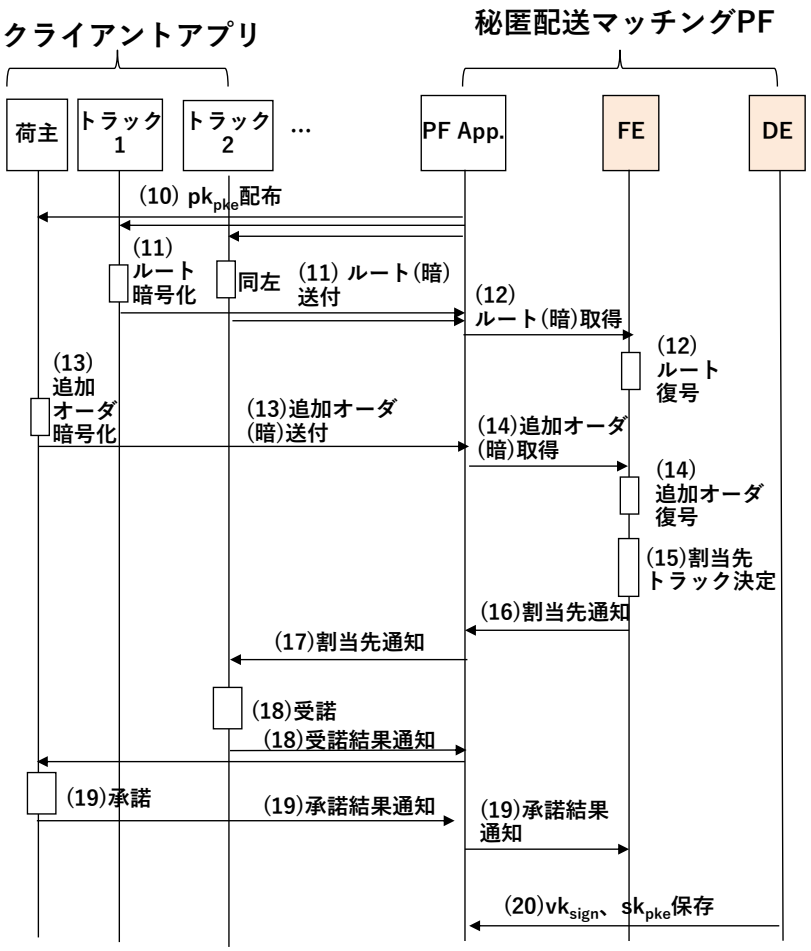

図 4-6 マッチングの処理フロー

(12) FE は PF App.からルートを取得し復号

(13)荷主は追加オーダを暗号化し PF App.に送付

(14) FE は PF App.から追加オーダを取得し復号

(15) FE はマッチング処理にて割り当て先トラックと辺を決定

(16) FE は割り当て先のトラックと辺を PF App.に通知

(17) PF App.は割り当て先のトラックに通知

(18)トラックは受諾するかを選択し，受諾結果を PF App.に通知，および PF App.が荷主に受諾結果・トラックを通知

(19)荷主が承諾結果を PF App.に通知し，PF App.は FE に承諾結果を通知

(20) DE は Sealing で sk_{pke} と vk_{sign} を保存（１日の処理終了後）

ここで（12）にて PF App.はルートを SGX のメモリ制約（第一世代の SGX では EPC (Enclave Page Cash) 96MB）の上限を超えない範囲で複数件まとめて送る．また（17）で

トラックに通知する際には，予め FE において追加オーダを sk_{pke} で暗号化してトラックに送付し，トラックは pk_{pke} で追加オーダを復号して，受諾するか否かを選択する．なお（18）（19）でトラックや荷主が承諾しなかった場合は，そのトラックのルート以外で追加距離が最短の辺を割り当て先として，（17）以降の処理を繰り返す．なお（19）で FE に承諾結果を通知するのは，一度マッチングが成立したルートは次のマッチングの割り当て候補から外すためである．

4.3.7　サービス提供者の不正抑止

ここで，FE を実行するサービス提供者が，FE の処理に不正を加えると秘密鍵や復号したルート，追加オーダを閲覧できる懸念がある．そこで，4.3.5 節で説明したように Key Manager が FE の処理内容の認証を行って sig_f を生成し，DE が FE を認証することで，秘匿配送マッチング PF では Key Manager が認証したプログラム以外は実行できないようにしている．

また Key Manager は，4.3.5 節の（4）～（6）を行う際に，秘匿配送マッチング PF に $mrenclave_f$ と合わせて FE のソースコードを提出させ，FE の処理に不正（例：SGX 内で平文にしたルートと追加オーダを OCALL で PF App.に出力する等）がないかを確認する想定である．また DE の処理についても，初回および処理内容を変更した際に Key Manager にソースコードを提出させて不正がないか確認する想定である．

4.4　評価環境の実装と評価

本節では，評価環境の実装および処理時間の評価と，サービス提供者に対する秘匿性の評価について述べる．

4.4.1　評価環境の実装

4.3 節で説明した秘匿配送マッチングの評価環境の実装を行った．開発に用いた PC や開発環境を表 4-5 に示す．評価環境では，Key Manager およびトラック，荷主は秘匿配送マッチング PF と同じ PC で実装している．また（17）以降の処理は PF の処理時間に与える影響は僅かと判断し評価対象外としている．（2）の R. A. については PF の処理とは切り分けて実装し，処理時間を評価している．

表 4-5 開発に用いた PC および開発環境

項目		仕様
PC	マシン名	HP 250G7 Notebook PC
	CPU	Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz
	SGX	第一世代 (EPC：96MB)
	8GB	8GB
	OS	Windows 10 Home
NW 環境	プロトコル	Wi-Fi 802.11n
	NW 帯域	2.4GHz
	リンク速度（送受信）	300 / 300 (Mbps)
開発環境	統合開発環境	Visual Studio Community 2017
	SGX の SDK	Intel SGX SDK ver 2.15.100.4
	言語	C++
	R.A.の種別	EPID 方式
用いた暗号	暗号ライブラリ	OpenSSL1.1.1m
	(pk_{pke}, sk_{pke})	RSA[82]-2048bit with OAEP SHA-256
	(vk_{sign}, sk_{sign})	RSA-3072bit

4.4.2 処理時間の評価方法

4.2.5 節で述べた, 289 件のトラックのルートが含まれる配送マッチング問題を 1 回あたり 3.6 秒以内に解く目標を満たせるか確認する．また，ルート件数を SGX のメモリサイズの上限を超えない最大 5 万件まで増やした場合の処理時間を評価する. 各トラックのルートには 4 つの辺が含まれるものと仮定する. これは例えば倉庫や集配所などの出発地から 3 地点を立ち寄って各地点で荷降ろしと荷積みを行い, 出発地に戻る場合のルートに該当する．処理時間の計測には Windows の Query Performance Counter を使用した．処理時間は 10 回処理を行った平均値を用いる.

ここで SGX では Stack と Heap のサイズについて，双方のサイズの和が EPC の上限を超えない範囲で変更ができる．Intel SGX SDK で設定されている Stack と Heap の初期値はそれぞれ 0.26MB と 1.0MB であるが，同じルート件数の場合に Stack と Heap のサイズを変えて処理を試行しても処理時間は殆ど変わらないことが分かった. また評価環境では Heap に平文にしたルートと追加オーダを保持しているため，ルート件数を増や

すと Heap の枯渇が起き，マッチング処理が行われなくなる．そこでルート件数を増やすに連れて Heap サイズを増大させて処理時間を評価する．なお，Stack は Enclave 内外のデータの授受に用いられるメモリで，FE の中にルート，オーダを送る際に使われているが，Stack サイズを変えても処理時間への影響はない．

4.4.3 処理時間の評価結果

評価結果を説明する．評価結果は，秘匿配送マッチング PF におけるマッチング処理（暗号化されたルートと追加オーダを FE 内に送り，割り当て先のトラックを導出する処理）と，マッチング処理よりも前に行う処理（鍵生成と FE，DE の認証およびルートと追加オーダの暗号化）に分けて説明する．

表 4-6 に，マッチング処理よりも前に行う処理とその実施頻度を示す．R.A.に約 4 秒掛かるが，本サービスの利用者の少ない夜間などに行えば良い処理のため問題ない．またルートと追加オーダの暗号化については，本来クライアント端末で行うが，いずれも所要時間は僅かである．なお R.A.の処理時間は，本章と同じ EPID 方式の場合，文献[81]では数 10msec，文献[82]では約 1.8 秒と報告されており，文献[82]にて処理時間の大部分が IAS への問い合わせに要する時間と報告されている．[81], [82]の R.A.の処理時間が本章の計測結果より短いのは，NW 環境（[81]は 1Gb イーサネット，[82]は詳細未公表であるが大学の学内 LAN を介して IAS に接続）や CPU の性能差（[81]は Intel Core i5-10400@4.3GHz, [82]は Intel Core i7-9700K@ 3.60 GHz）によるものと考える．

次に，表 4-7 にルート件数と秘匿配送マッチング PF におけるマッチング処理の所要時間の関係を示す．ルート件数が 1,000 件以下の場合は Stack と Heap とも 8.4MB に設定し，ルート件数が 1 万件，5 万件の時は Heap の枯渇が起きないように Heap サイズを設定した．

ルート件数がいずれの場合でも，FE 内で行うルートの復号に掛かる時間が支配的であり， 1 ～3 の処理全体の 99%以上を占めるが，ルートが 289 件の場合でも 1 ～3 の処理を合わせて約 2.5 秒でマッチングができ，4.2.5 節で示した 3.6 秒以内という制約を満たす．またルート件数とルートの復号に掛かる処理時間の関係を図 4-7 に，ルート件数と割り当て先トラック決定に掛かる処理時間の関係を図 4-8 にそれぞれ示す．図 4-7，図 4-8 とも右側のグラフはルート件数 0 付近を拡大したグラフである．ルートの復号に掛かる処理時間および割り当て先トラック決定の処理時間とも，ルート件数に対してほぼ線形で増えることが分かる．

表 4-6 マッチング処理よりも前に行う処理の処理時間

項目	処理時間（msec)	実施頻度
鍵の生成	700	DE の処理変更時
DE の認証（R. A.）	4,065	１日１回程度
sig_fの生成	4	FE の処理変更時
DE による FE の認証（L.A.）	4	１日１回程度
オーダの暗号化	0.2	毎回のマッチング時
ルートの暗号化（1 件あたり）	4	毎回のマッチング時
合計	4,777.2	-

表 4-7 秘匿配送マッチング PF におけるマッチング処理の処理時間

ルート件数	100	289	1,000	10,000	50,000
Stack Size (MB)	8.4	8.4	8.4	8.4	0.65
Heap Size (MB)	8.4	8.4	8.4	50.3	86.0
1.ルートの取得と復号(ms)	706.6	2459.0	7,398.3	76,515.9	346,858.0
2.オーダ取得・復号（ms)	2.1	1.8	1.6	1.7	1.6
3.割り当て先トラック決定(ms)	0.5	2.2	3.6	20.4	172.7
1～3 合計（ms)	709.2	2,463.0	7,403.5	76,524.0	347,032.3
2・3 の合計（ms)	2.6	4.0	5.2	22.1	174.3

なお Stack と Heap のサイズは，ルート件数 5 万件の時に設定した合計 86.65MB が上限であり，それ以上のサイズを割り当てると Enclave の起動ができなくなる．ただしこれは Windows 環境下で第一世代の SGX を用いた場合の事象であり，Linux 環境下では処理速度が低下する代わりに，より大きなメモリサイズを設定できることが知られている．なお第二世代の SGX では上限の EPC サイズを動的に変更でき，ルート件数や各トラックの辺の数をさらに増やすことが可能と考えられる．

ここで，評価環境ではルートを全件まとめて SGX の中に送り復号して，それが完了したら荷主の追加オーダを SGX に送って復号後にマッチングをしている．一方，実際

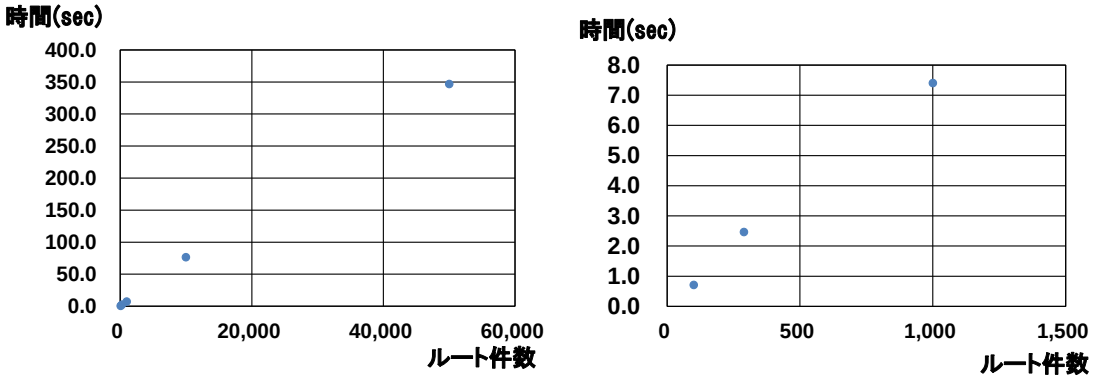

図 4-7 ルート件数とルートの復号に掛かる時間の関係

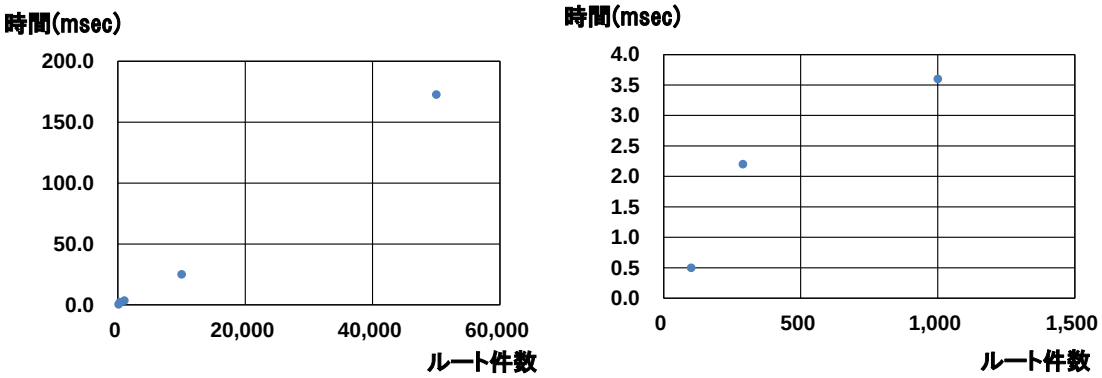

図 4-8 ルート件数と割り当て先トラック決定処理の時間の関係

のサービスにおいては秘匿配送マッチング PF にルートが到着したら順次予め SGX の中に送って事前に復号しておけば良く，荷主にとっての応答時間は表 4-7 の 2 と 3 の合計に通信時間を加えた時間と考えることができる．すると，ルート件数が 5 万件の場合でも，荷主にとっては 174.3msec で応答が返って来ることから，十分高速なマッチングができている．なお本来は荷主端末と PF との通信時間を考慮する必要がある．荷主から秘匿配送マッチング PF には表 4-2 に示す追加オーダを RSA-2048bit で暗号した暗号文を送信し，また秘匿配送マッチング PF から荷主にはマッチング結果を送信するが，ともにデータサイズは数 kB であるため，通信時間は僅かと考える．

4.4.4 秘匿性に関する考察

本節では，サービス提供者，トラック，荷主に対してルート・追加オーダの秘匿が実現できているか考察する．表 4-8 に想定されるルートと追加オーダの閲覧方法と，それに対して検討した対策を示す．まず秘密鍵 sk_{pke} を Key Manager から DE に提供する際に

表 4-8 サービス提供者，荷主，トラックのルートと追加オーダの閲覧方法と対策

閲覧方法	対策
Key Manager から DE に sk_{pke} を渡す際に入手	・R.A.による秘密鍵 sk_{pke} の提供 （AES 共通鍵で暗号化）
FE, DE の処理を改ざん	・Key Manager による DE の認証 ・Key Manager による FE の署名と DE による FE の認証 ・ソースコード変更時の KeyManager の確認

サービス提供者，トラック，荷主が不正入手する可能性が有る．これについては，SGX では R.A.を行った際に，ECDH 鍵交換が行われ AES 共通鍵が生成される[82][83]．この共通鍵で sk_{pke} を暗号化した後に Key Manager から DE に渡すことで，サービス提供者の sk_{pke} の入手が抑止できる．なお[82]ではこの共通鍵は Enclave 外から入手不可能と述べているが，DE のソースコードをサービス提供者，トラック，荷主が改変すれば入手できる可能性が有る．しかし，そうすると $mrenclave_{DE}$ の値が変わって Key Manager が保持している正解値と一致せず DE の認証が通らなくなるため，DE のソースコードの改変は不可能である．

次に，サービス提供者，トラック，荷主が FE，DE のソースコードを改ざんして秘密鍵 sk_{pke} を入手する，Enclave の中から外にデータ等を出力可能な OCALL などによってルート・追加オーダを閲覧できるという可能性が有る．これに対して，まず DE については，R.A.を行う際に Key Manager は $mrenclave_{DE}$ で DE の処理の不変性を検証しているため，DE の改ざんを防ぐことができる．また FE についても Key Manager が処理内容を署名し，さらに DE が L. A.で FE を認証する際に sig_f を検証しているため FE の改ざんを防ぐことができる．ここで，サービス提供者，荷主，トラックが秘密鍵 sk_{pke} を入手する以外の方法で Enclave 内のデータを見るには，ECALL（Enclave 外から中の処理を呼ぶ関数）か OCALL（Enclave の中から外の処理を呼ぶ関数）を用いる必要が有り，ECALL, OCALL のいずれも Enclave の中と外の双方のソースコードに実装する必要がある．すると，$mrenlcave_{DE}$ または $mrenclave_f$ の値が変わって，FE または DE の認証が通らなくなるため，実質的に ECALL, OCALL の実装は不可能である．

よって，サービス提供者，荷主，トラックが FE, DE の中の処理を改ざんしてルート，追加オーダを閲覧することはできない．なお，サービス提供者が FE，DE の処理を変更する正当な理由がある場合は Key Manager がソースコードを確認することで，不正が無いことを確認する想定である．

4.4.5　他の秘密計算技術との比較

本節では TEE と他の秘密計算技術との比較を行い，秘匿配送マッチングに TEE 活用が適切であったか検証する．まず，準同型暗号では 4.2.6 節で述べた処理時間の課題がある上に，暗号で演算を行うため，TEE のようにトラックが増えた場合に事前にルートを復号しておいて処理時間を削減することができない．また，全ての演算が実施可能ではないため，今後配送時間や積載量の制約を加味するなどしてマッチングアルゴリズムが複雑になった場合に対応できない可能性がある．次に準同型暗号よりも高速な手法として，複数サーバに入力データを断片（シェア）分散して秘密計算を行う秘密分散[84]があるが，追加距離計算における乗算の処理などにおいてサーバ間の通信のオーバヘッドが発生し，やはりトラックが増えると処理速度が課題になると考える．

一方 TEE を用いた場合，TEE 内部で平文による処理を行うため，準同型暗号，秘密分散に比べて，マッチングアルゴリズムの拡張性も高く高速である．またアルゴリズムを拡張した場合の追加の処理時間は平文と同等と考えることができ，処理時間を見積もりやすい．このように，秘匿配送マッチングで求められる高速性と拡張容易性の面で TEE は準同型暗号と秘密分散よりも優れており，現時点では秘匿配送マッチングに TEE を用いることが適切である．

4.5　まとめ

サービス提供者を含む他者に対してルートと追加オーダを秘匿した状態で配送マッチングを実現する秘匿配送マッチングにおいては処理時間の課題があり，準同型暗号では制約を満たせない．一方関数型暗号を TEE で実現すれば高速化が可能であることは知られていたが，TEE の中でルートと追加オーダを平文に戻す際のオーバヘッドがあるため，処理時間の制約を満たせるかは不明であった．

本章では，IRON[20]のプロトコル（システム構成・処理手順）をベースに，SGX を用いた多入力関数型暗号を実現し，評価環境を構築して処理時間を評価した．その結果トラックのルートが 289 件存在するマッチングを約 2.5 秒で実施でき，3.6 秒以内という制約を満たせることを確認した．また，ルートと追加オーダが秘匿できることを確認し，秘匿配送マッチングにおける TEE を用いた関数型暗号の有効性を確認した．ここで処理時間は SGX メモリサイズの上限内でトラックのルート数に対して線形で増加することが知見として得られた．またルートと追加オーダの復号に掛かるオーバヘッドが，PF のマッチング処理時間の 99%以上を占めることが分かった．ただし秘匿配送マッチング

PF では受信したトラックのルートを随時 TEE 内に送り復号して保持しておくことで，ルート件数が増えても高速なマッチング処理が実施でき，荷主に対する応答時間を十分に短くできる見通しである．

今後の課題として，SGX ではサイドチャネル攻撃からの脆弱性が指摘されており，良く知られているものとして，プログラムの中身や実行順序をメモリの配置される場所から推測して情報を得るキャッシュタイミング攻撃がある[53]．対策として，文献[55][57]のように ORAM (Oblivious RAM) を用いてメモリやディスク上に保管したデータへのアクセスパタンを秘匿する方法が一般的であるが，計算時間が大きく増加する懸念がある．それに対して，Costan らはメモリアクセスパタンが秘匿可能な TEE を実現した Sanctum を開発しており，処理時間のオーバヘッドは通常の TEE の 10%程度である[86]．今後本研究でも同 TEE の適用を検討する．その他，Key Manager やクライアント端末，通信路のセキュリティ，および PF における秘匿性以外のセキュリティの担保方法についても検討が必要である．また現在は追加距離最小という指標でのみ割り当て先トラックを決定しているが，積載量や時間制約などの他の指標を加味したマッチングについても検討する．

第5章
結論

5.1　本研究のまとめ

本論文では，物流情報システムにおける業務効率化とセキュリティ強化を提案し，これらの研究成果を以下の 4 章に分けて報告した．

第 1 章では，物流業務において主に倉庫管理，追跡管理，配送管理に関する各機能を提供する物流情報システムの中で，特に倉庫管理が対象とする倉庫業務の効率化と追跡管理のトレーサビリティおよび配送管理の配送マッチングのデータ秘匿性に関するセキュリティ強化が必要であることを述べた．その上で，(1) 混載 PL を含めた PL の格納期間を加味した倉庫内在庫の格納先決定方法の確立，(2)トレーサビリティ管理における秘匿性と BC 書き込み性能の両立，(3)秘匿配送マッチングを TEE で実現した際の処理時間の評価 の 3 つの課題が重要であることを示した．更に各課題に対して，(1) 格納期間を加味した在庫格納先決定方式の提案，(2) 分割 RH 方式による BC 利用トレーサビリティ管理システムの提案，(3) TEE を用いた関数型暗号による秘匿配送マッチングシステムの処理時間の評価 という研究方針を定めた．

第 2 章では，格納 PL 上の各商品の過去の出庫実績を評価し，出庫タッチ数が多く格納期間が短いと予測される格納 PL を優先度高エリア段に格納および在庫移動を行う格納先決定方式 SPLP を提案した．SPLP では混載も考慮して出庫タッチ数と格納期間から PL の評価値を生成し，倉庫内に格納済の PL である在庫 PL の評価値との順位比較に基づいて PL の保管エリア格納時の格納先ロケを決定する．また格納済在庫 PL の中で評価値が高い PL を優先度 1 エリア段に在庫移動を行う．倉庫のデータを用いたシミュレーションによって，既存手法として出庫タッチ数のみを考慮して格納先エリア段を決定し在庫移動は行わない方法に比べ，SPLP は在庫移動無しの場合 4.40%，在庫移動有りの場合 1 日 10PL の移動を行うことで，在庫移動工数を含めて 5.65%出庫工数を削減できることを示した．

第 3 章では，BC による耐改ざん性に加えて，書き込み性能とデータ秘匿性を両立する分割 RH 方式によるトレーサビリティ管理システムを提案した．提案の分割 RH 方式では，商品の製造元メーカごとに TR.データを分割することでデータ秘匿性を保証し，またその複数のハッシュ値から生成した分割 RH を BC の 1Tx に複数書き込むことで書

き込み性能を担保する． BC 基盤として Hyperledger Fabric を用いて書き込み性能の評価環境を構築し，BC サーバ 2 台，BC クライアント 5 台の構成の評価環境にて，コンビニ電子タグ 1,000 億宣言で想定される年間 1,000 億商品を取り扱い可能であることを示した．また，耐改ざん性および商品混載時のデータの秘匿性の要件を満足できることを確認した．

第 4 章では，TEE として SGX を用いて関数型暗号を実現し，秘匿配送マッチングシステムの評価環境を構築し，処理性能を評価した．その結果，トラックのルートの復号に掛かる処理が支配的であるが，目標である 289 台のトラックのルートが含まれる 1 回のマッチングを 2.5 秒（目標 3.6 秒）で処理可能であることを確認した．また，処理時間は SGX メモリサイズの上限内でトラックのルート数に対して線形で増加することが知見として得られ，サービス提供者を含む他者にルートと追加オーダが秘匿できることを確認した．

本研究では，物流情報システムにおける業務効率化とセキュリティ強化に関する 3 つの課題を解決した．本研究の成果により，倉庫内の出庫工数を削減する在庫配置が実現できる．また，複数ステークホルダのデータが関連するトレーサビリティ管理および配送マッチングにおいて，処理性能を担保しつつデータのセキュリティの強化が図れる．

5.2 今後の課題

最後に，本研究における今後の課題について述べる．本論文では，(1) 混載 PL を含めた PL の格納期間を加味した倉庫内在庫の格納先決定方法の確立，(2) トレーサビリティ管理における秘匿性と BC 書き込み性能の両立，(3) 秘匿配送マッチングを TEE で実現した際の処理時間の評価 という 3 つの課題に対して，第 2 章〜第 4 章で解決方法と，個々の解決方法における今後の課題を示した．今後は，これらの解決方法を実際の業務に適用して，その効果や影響を評価する必要がある．

また，その他の課題としては以下のものがある．

(1) 物流情報システムの各機能を掛け合わせた業務効率化の効果向上とシステム全体のセキュリティ強化

本論文では，在庫配置の改善による業務効率化向上と，トレーサビリティおよび配送マッチングのセキュリティ強化を対象としたが，今後物流情報システムの利用者を拡大する上では，各機能を掛け合わせた業務効率化向上とシステム全体のセキュリティ強化が求められる．例えば倉庫管理では作業員が複数ロケを巡回してピックする際に，在庫

配置を改善した上で出庫時の作業計画である巡回経路や出庫オーダ内の出庫商品の割り付けを最適化することや, 配送管理では配送マッチングを行った後にトラックのルート全体が最短となるように配送経路を最適化することが挙げられる. セキュリティの観点では, 倉庫で扱う商品情報（商品名, 入出荷先など）を物流情報システムの管理者（サービス提供者）に秘匿した状態で, 倉庫の作業計画や在庫配置を立案することなどが挙げられる.

(2) 拠点間のデータ共有と需要予測による SC 全体での在庫最適化

本論文では, 個々の倉庫を対象に格納期間を加味した格納先決定方式を開発し, 出庫工数削減を実現した. 一方で, 物流情報システムにおいて, 製造から販売までの各拠点（工場・倉庫・小売店など）の在庫データや製造・出荷・販売実績を共有し, またそれらのデータを用いて各拠点の出荷・販売量の需要を予測することで, 工場・倉庫に存在する余剰在庫の削減や店舗における欠品の抑止を図り, SC 全体での在庫最適化を行う必要がある. その際に, 販売金額や出荷・販売に関わった具体的な企業名などは必要に応じて秘匿することが望ましい.

(3) マルチモーダルによる中継・共同配送のマッチングとトレーサビリティ管理のスコープ拡大

本論文では, 荷主がトラックを手配できない時などに活用可能な秘匿配送マッチングの方法を確立した. しかし物流業界ではトラック不足が深刻であることから, トラック以外の鉄道・船舶等の輸送手段（モーダル）も活用したマルチモーダルの配送が行われている. また長距離配送の場合, 複数のトラック（配送業者）を乗り継いで配送を行う中継配送や, 同じ方面に配送する際には荷主間でトラックをシェアリングする共同配送が行われている.

そこで主に秘匿配送マッチングの技術を拡張して, 機微情報を他者に秘匿しつつ, 中継・共同配送も加味して荷主の追加オーダを, 適切なモーダルに割り当てを行う高度なマッチング技術の開発が必要である. また, 輸送に関わるステークホルダが増えるに伴い, 輸送中の商品の品質管理（温度管理, 振動・衝撃の有無など）に対するニーズが増える. 本論文ではトレーサビリティ管理の対象をコンビニ商品としていたが, トレーサビリティ管理システムではそれ以外の商品の TR.データも扱えるよう, 更なる処理性能の向上などに取り組む必要がある.

謝　辞

本研究の全過程を通じて，終始懇切丁寧なるご指導とご鞭撻を賜りました大阪大学 大学院情報科学研究科 マルチメディア工学専攻 藤原融招へい教授（島根大学教授，大阪大学名誉教授），矢内直人准教授，薦田憲久招へい教授（大阪大学名誉教授）に深く感謝申し上げます.

本研究をまとめるにあたり，貴重なお時間を割いて頂き，丁寧なるご教示を賜りました大阪大学 大学院情報科学研究科 マルチメディア工学専攻 伊達進教授，前川卓也准教授に厚く感謝申し上げます.

本研究をまとめるにあたり，親切なるご助言を賜りました大阪大学 大学院情報科学研究科 マルチメディア工学専攻 原隆浩教授，鬼塚真教授，松下康之教授に深く感謝申し上げます.

本研究の機会と大学院博士後期課程に進学する機会を与えて頂くとともに，研究を進めるにあたりご配慮を賜りました（株）日立製作所 研究開発グループ サービスシステムイノベーションセンタ長 谷繁幸博士に心より御礼申し上げます.

直属の上司として会社業務との両立をご支援頂いた，（株）日立製作所 研究開発グループ DX エンジニアリング研究部 UL 主任研究員 足立哲朗氏，元 UL 主任研究員 齊藤元伸氏（現 Hitachi Asia Co., Ltd. Chief Researcher），デジタルエコノミー研究部 UL 主任研究員 宮田克也氏に厚く御礼申し上げます．また筆者の大学院博士課程在学中に会社業務をサポートして頂いた，DX エンジニアリング研究部の関係各位に厚く御礼申し上げます.

第 2 章の研究に関して，研究の機会を与えて頂くとともに，多大なるご支援を頂きました（株）日立物流（現ロジスティード（株）） 嶋津泰毅氏，植木隆雄氏を始めとする同社の関係者の皆様に心より感謝いたします．また，共同研究者として様々なご討論，ご助言を頂いた（株）日立製作所 研究開発グループ 元主任研究員 荒宏視博士（現（株）ハピネスプラネット 取締役），UL 主任研究員 永原聡士氏，研究員 末光一成氏に厚く御礼申し上げます.

第 3, 4 章の研究に関して，共同研究者として様々なご討論やご助言を頂きました（株）日立製作所 研究開発グループ 主任研究員 小坂忠義氏，長谷川学氏に厚く御礼申し上げます．また第 3 章の実験環境構築にご支援をいただいた，元（株）日立ソリューションズ 松岡幸典氏，（株）日立情報通信エンジニアリング 青木優香氏に深く感謝申し上げます.

最後に，大学院博士後期課程在学中に常に支えてくれた妻と，学位取得を応援してくれた娘，両親，妹に感謝いたします.

参考文献

[1] 国土交通省：「総合物流施策大綱 (2021-25)」(2021), https://www.mlit.go.jp/seisakutokatsu/freight/butsuryu03100.html (2023/2/26 閲覧)

[2] 富士電機（株）:「物流・倉庫部門における人手不足の実態調査」(2021), https://www.fujielectric.co.jp/products/logistics/research/research09/box/pdf/research_2021_09_busturyuhitodebusoku.pdf (2023/2/26 閲覧)

[3] 内閣府科学技術・イノベーション推進事務局：「SIP スマート物流サービス研究開発計画」(2022), https://www8.cao.go.jp/cstp/gaiyo/sip/keikaku2/11_logistics.pdf (2023/2/26 閲覧)

[4] 石川和幸:「エンジニアが学ぶ物流システムの「知識」と「技術」」, (株) 翔泳社 (2018)

[5] デジタル庁：「包括的データ戦略」(2021), https://www.digital.go.jp/assets/contents/node/information/field_ref_resources/576be222-e4f3-494c-bf05-8a79ab17ef4d/210618_01_doc03.pdf (2023/2/26 閲覧)

[6] Coyle, J. J., Bardi, E. J., and Langley, C. J. : "*The Management of Business Logistics : A Supply Chain Perspective*," South Western Education Publishing (2002)

[7] ロジ・ソリューション（株）出版プロジェクト：「図解でわかる 物流の基本と仕組み」, （株）アニモ出版 (2015)

[8] Reyes, J., Sorano-Charris, E. and Montoya-Torres, J. : "The Storage Location Assignment Problem: A Literature Review," *International Journal of Industrial Engineering Computations*, Vol.10, No.2, pp.199-214 (2019)

[9] 國領次郎 : "トレーサビリティとシェアリングエコノミーの進化," 研究 技術 計画, Vol.32, No.2, pp.105-116 (2017)

[10] Nakamoto, S. : "Bitcoin: A Peer-to-Peer Electronic Cash System," *Decentralized business review 2008* (2008), https://assets.pubpub.org/d8wct41f/31611263538139.pdf (23/2/26 閲覧)

[11] （株）三菱総合研究所：「ブロックチェーン」,「ICT によるイノベーションと新たなエコノミー形成に関する調査研究」報告書, pp.87–90 (2018)

[12] IBM："About IBM Food Trust" (2019)

[13] 岡部達哉，三谷陽，徐昕，坂本快矢統，水摩智，並木陽彦，黄浩倫，田村由佳：“ブロックチェーン技術を用いた車両データ・製品トレーサビリティデータの改ざん防止,” *DENSO TECHNICAL REVIEW*, Vol.24, pp.42–52 (2019)

[14] 国立研究開発法人新エネルギー・産業技術総合開発機構：「IoT を活用した新産業モデル創出基盤整備事業 IoT の社会実装推進に向けて解決すべき新規課題に関する システムの開発 電子タグを用いたサプライチェーンの 情報共有システムの構築に関する研究開発 成果報告書」 (2018), https://www.meti.go.jp/policy/economy/distribution/smartsupplychain/FY29RFIDreport.pdf (2023/3/5 閲覧)

[15] Androukaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., Caro, A. D., *et al.*：“Hyperledger Fabric : A Distributed Operating System for Permissioned Blockchains,” in *Proceedings of European Conference on Computer Systems* (EuroSys '18), Vol.30, pp.1–15 (2018)

[16] 尾根田倫太郎，秋田佳記，何岩彬，竹林陽，小倉拓人，鈴木貴之：“ブロックチェーン基盤性能検証 ー Hyperledger Fabric, Quorum, Ethereum の横串比較ー,” 情報処理学会論文誌デジタルプラクティス, Vol.10, No.3, pp.506–523 (2019)

[17] 光成滋生：「クラウドを支えるこれからの暗号技術」，（株）秀和システム (2015)

[18] 須崎有康：“Trusted Execution Environment の実装とそれを支える技術,” 電子情報通信学会 基礎・境界ソサイエティ Fundamentals Review, Vol.14, No.2, pp.107–117 (2020)

[19] Boneh, D., Sahai, A., and Waters, B.：“Function Encryption：Definitions and Challenges,” *Cryptology ePrint Archive*, Paper 2010 / 543 (2010)

[20] Fisch, B., Vinayagamurthy, D., Boneh, D., and Gorbunov, S.：“IRON : Functional Encryption Using Intel SGX,” in *Proceedings of Conference on Computer and Communications Security*, pp.765–782 (2017)

[21] Pinto, S. and Santos, N.：“Demystifying Arm TrustZone : A Comprehensive Survey,” *ACM Computing Surveys,* Vol. 51, Issue 6, No.130, pp.1–36 (2019)

[22] Costan, V., and Devadas, S.：“Intel SGX Explained,” *IACR Cryptol. ePrint Archive 2016*, pp.86－204 (2016)

[23] Gu, J., Goetschalckx, M., and McGinnis, L. F.：“Research on Warehouse Operation: A Comprehensive Review,” *European Journal of Operation Research*, Vol.177, No.1, pp.1–21 (2007)

[24] Koster, R. D., Le-Duc, T., and Roodbergen, K. J. : "Design and Control of Warehouse Order Picking : A Literature Review," *European Journal of Operation Research*, Vol.182, No.2, pp.481–501 (2007)

[25] Graves, S. C., Hausman, W. H., and Schwarz, L. B. : "Storage-Retrieval Interleaving in Automatic Warehousing Systems," *Management Science*, Vol.23, pp.935–945 (1977)

[26] Jarvis, J. M. and McDowell, E. D. : "Optimal Product Layout in an Order Picking Warehouse," *IIE Transactions*, Vol.23, pp.93–102 (1991)

[27] Chuang, Y. F., Lee, H. T., and Lai, Y. C. : "Item-Associated Cluster Assignment Model on Storage Allocation Problems," *Computers & Industrial Engineering*, Vol.63, No.4, pp.1171–1177 (2012)

[28] Mantel, R. J., Heragu, S. S., and Schuur, P. C. : "Order Oriented Slotting : A New Assignment Strategy for Warehouses," *European Journal of Operation Research*, Vol.1, No.3, pp.301–316 (2007)

[29] 南賢一，飯塚博幸，古川正志，山本雅人："ネットワークトポロジーを考慮した自己組織化マップによる商品配置方法の提案," 人工知能学会論文誌, Vol.30, No.6, pp.737–744 (2015)

[30] Wutthisirisart, P., Noble, J. S., and Chang, C. A. : "A Two-Phased Heuristic for Relation-based Item Location," *Computers & Industrial Engineering*, Vol. 82, pp. 94–102 (2015)

[31] Goetschalckx, M., and Ratliff, H. D. : "Shared Storage Policies Based on the Duration Stay of Unit Loads," *Management Science*, Vol.36, No.9, pp.1120–1132 (1990)

[32] Jaikumar, R., and Solomon, M. M. : "Dynamic Operational Policies in an Automated Warehouse," *IIE Transactions*, Vol.22, No.4, pp.370–376 (1990)

[33] Kofler, M., Beham, A., Wagner, S., Affenzeller, M., and Achleitner, W. : "Re-warehousing vs. Healing : Strategies for Warehouse Storage Location Assignment," in *Proceedings of 3rd IEEE International Symposium on Logistics & Industrial Informatics*, pp.77–82 (2011)

[34] Kofler, M., Beham, A., Wagner, S., Affenzeller, M., and Reitinger, C. : "Reassigning Storage Locations in a Warehouse to Optimize the Order Picking Process," in *Proceedings of 22nd European Modeling & Simulation Symposium (EMSS 2010)*, pp.77–82 (2010)

[35] Chen, L., Langevin, A., and Riopel, D. : "A Tabu Search Algorithm for the Relocation Problem in a Warehousing System," *International Journal of Production Economics*, Vol.129, pp.147–156 (2011)

[36] 橋本英樹，野々部宏司：“入門タブー探索法,” オペレーションズ・リサーチ ： 経営の科学， Vol. 58, No. 12, pp. 703-707 (2013)

[37] Chen, G., Feng, H., Luo, K., and Tang, Y. : “Retrieval-Oriented Storage Relocation Optimization of an Automated Storage and Retrieval System,” *Transportation Research Part E : Logistics and Transportation Review*, Vol.155, 102508 (2021)

[38] Powell, W. B. : ”Approximate Dynamic Programming : Solving the Curses of Dimensionality (Wiley Series in Probability and Statistics),” Wiley (2007)

[39] 富士通（株） ：「Picking Optimizer」 https://www.fujitsu.com/jp/solutions/industry/logistics/product/pickingoptimizer/ （23/6/24閲覧）

[40] 林建太朗，伊呂原隆，佐々喜仁：“需要に基づく商品分類を考慮したピッキング形態”，日本経営工学会論文誌, Vol. 68, No. 1, pp. 33–46 (2017)

[41] GS1 ：“GS1 General Specifications ver.23, ” pp.212-219 (2023)

[42] GS1 :「EPCIS 及び CBV 導入ガイドライン」(2017)

[43] Staake, T., Thiesse, F., and Fleisch, E. : “Extending the EPC Network : the Potential of RFID in Anti-Counterfeiting,” in *Proceedings of 2005 ACM Symposium on Applied Computing*, pp.1607–1612 (2005)

[44] Gavin, W. : “Ethereum: A Secure Decentralised Generalised Transaction Ledger,” *Ethereum project yellow paper* 151. 2014, pp.1-32 (2014)

[45] Westerkamp, M., Victor, F., and Küpper, A. : “Blockchain-Based Supply Chain Traceability : Token Recipes Model Manufacturing Processes,” in *Proceedings of 2018 IEEE Conference on Internet of Things*, pp.1595–1602 (2018)

[46] Toyoda, K., Mathiopoulos, P. T., Sasase, I. and Ohtsuki, T. : “A Novel Blockchain-Based Product Ownership Management System (POMS) for Anti-Counterfeits in the Post Supply Chain,” *IEEE Access*, Vol.5, pp.17465–17477 (2017)

[47] Lin, Q., Wang, H., Pei, X. and Wang, J. : “Food Safety Traceability System Based on Blockchain and EPCIS,” *IEEE Access*, Vol.7, pp.20698–20707 (2019)

[48] Shakhbulatov, D., Arora, A., Dong, Z., and Rojas-Cessa, R. : “Blockchain Implementation for Analysis of Carbon Footprint across Food Supply Chain,” in *Proceedings of 2019 IEEE Blockchain*, pp.546–561 (2019)

[49] Tan, B., Yan, J., Chen, S. and Liu, X. : “The Impact of Blockchain on Food Supply Chain: The Case of Walmart,” in *Proceedings of SmartBlock 2018. Lecture Notes in Computer Science*, Vol.11373, pp.167–177 (2018)

[50] Master Card and Visa：“*SET Secure Electronic Transaction Specification Book 1, Book2, Book3”* (1997)

[51] 林卓也：“準同型暗号を用いた秘密計算とその応用,” システム/制御/情報，Vol.63, No.2, pp. 64–70 (2019)

[52] Goldwasser. S., Gordon S. D., Goyal. V., Jain. A., Katz. J., Liu. F., Sahai. A., Shi. E., and Zhou. H. S. : “Multi-Input Functional Encryption,” in *Proceedings of EUROCRYPT2014*, pp. 578–602 (2014)

[53] 加納英樹，Tibouchi. M.，阿部正幸 : “Intel SGX を用いた関数型タイムリリース暗号,” 暗号と情報セキュリティシンポジウム予稿集，No.2A3-5, pp.1–7 (2019)

[54] Bhatotia, P., Kohlweiss, M., Martinico, L., and Tselekounis, Y. : “Steel : Composable Hardware-based Stateful and Randomized Functional Encryption,” in *Proceedings of IACR International Conference on Public-Key Cryptography*, pp.709–736 (2021)

[55] 岩田大輝，清水佳奈 : “Intel SGX を用いた個人ゲノム情報解析システム,” 暗号と情報セキュリティシンポジウム予稿集，No. IC2-1, pp. 1–8 (2020)

[56] Mandal, A., Mitchell, J. C., Montgomery, H., and Roy, A. : “Data Oblivious Genome Variants Search on Intel SGX,” In *Proceedings Data Privacy Management, Cryptocurrencies and Blockchain Technology*, pp.296–310 (2018)

[57] Stefanov, E., Dijk, M., Shi, E., Chan, T., H., Fletcher, C., Ren, L., Yu, X., and Devadas, S. : “Path ORAM：An Extremely Simple Oblivious RAM Protocol,” *Journal of ACM*, Vol.65, No.4, pp.1-26 (2018)

[58] Naor, M., and Nissim, K. : “Communication Preserving Protocols for Secure Function Evaluation,” in *Proceedings of 33rd annual ACM symposium on Theory of computing*, pp.590-599 (2001)

[59] Canetti, R., Ishai, Y., Kumar, R., Reiter, M. K., Rubinfeld, R., and Wright, R. N. : “Selective Private Function Evaluation with Applications to Private Statistics,” in *Proceedings of 20th annual ACM symposium on Principles of distributed computing*, pp.293-304 (2001)

[60] Felsen, S., Kiss, Á., Schneider, T., and Weinert, C. : “Secure and Private Function Evaluation with Intel SGX,” in *Proceedings of 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop*, pp.165–181 (2019)

[61] Yao, A. : “Protocols for Secure Computations (Extended Abstract),” in *Proceedings of IEEE FOCS ‘82*, pp.160–164 (1982)

[62] Goldreich, O., Micali, S., and Wigderson, A. : “How to Play Any Mental Game,” in *Proceedings of 19th ACM Symposium on Theory of Computing*, pp.218–229 (1987)

[63] 古家直樹，永原聡士，末光一成，植木隆雄，荒宏視，嶋津泰毅 : “在庫格納期間を加味した逐次処理型の倉庫内格納先推奨技術の開発,” 日本経営工学会 2017 年秋季大会予稿集，pp.160-161 (2017)

[64] 古家直樹，永原聡士，末光一成，植木隆雄，荒宏視，嶋津泰毅 : “在庫格納期間を加味した逐次処理型の倉庫内格納先推奨技術の開発,” 日本経営工学会論文誌, Vol.70, No.2, pp.82-93 (2019)

[65] Furuya, N., Komoda, N., and Fujiwara, T. : “Improvement of Area Priority Setting of Sequential Processing Storage Location Recommendation Technology Considering Stock Storage Period,” in *Proceedings of 26th International Conference on Production Research 2021 (ICPR2021)*, No.3, pp.1-6 (2021)

[66] 古家直樹，末光一成，薦田憲久，藤原融 : “格納期間評価による倉庫内の在庫移動方式,” 日本経営工学会論文誌, Vol.73, No.4, pp.210-221 (2023)

[67] 古家直樹，長谷川学，小坂忠義，薦田憲久 : “分割ルートハッシュ方式を用いたブロックチェーン利用トレーサビリティ管理システム,” 電気学会論文誌 C, Vol.141, No.10, pp.1101-1114 (2021)

[68] 古家直樹，長谷川学，小坂忠義，薦田憲久 : “分割ルートハッシュ方式を用いたブロックチェーン利用トレーサビリティ管理システム,” 電気学会情報システム研究会予稿集, pp.85-90 (2020)

[69] 長谷川学，古家直樹，小坂忠義 : “データ秘匿性を考慮した物流トレーサビリティデータ検索技術の開発,” 電気学会情報システム研究会予稿集, pp.97–101 (2020)

[70] 古家直樹，矢内直人，藤原融 : “関数型暗号を用いた秘匿配送マッチングの TEE による実現,” 情報処理学会第 99 回 CSEC・第 49 回 SPT・第 98 回 EIP 合同研究発表会予稿集，2022-CSEC-99 (1), pp.1–8 (2022)

[71] Becker, G. : “Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis,” *Technical Report of Ruhr-University Bochum*, p.16 (2008), https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=d7c3aa65bc5df32d94dcc8b29dceca240bdf8bef (23/2/26 閲覧)

[72] Harman, P. : “Distributed Trusted Timestamping,” *Master thesis of Faculty of Informatics*, *Masaryk University* (2019), https://is.muni.cz/th/bgmmw/Trusted_Timestamping_v3_is.pdf (23/2/26 閲覧)

[73] 早川勝監修 :「ブロックチェーンの革新技術 Hyperledger Fabric によるアプリケーション開発」,（株）リックテレコム (2018)

[74] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION : "*Secure Hash Standard (SHS)* " (2012), https://csrc.nist.gov/csrc/media/publications/fips/180/4/final/documents/fips180-4-draft-aug2014.pdf (23/2/26 閲覧)

[75] 山田仁志夫, 大島訓 : "Hyperledger Fabric 1.0 概要," 第 1 回 Hyperledger Tokyo Meetup (2017), https://www.slideshare.net/Hyperledger_Tokyo/hyperledger-fabric-10 (23/2/26 閲覧)

[76] Thakkar, P., Nathan, S., Viswanathan, B. : "Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform," in *Proceedings of 2018 IEEE International Symposium on Modeling*, *Analysis, and Simulation of Computer and Telecommunication Systems*, pp.264–276 (2018)

[77] 倪永茂 : "ハッシュ法の実装と応用," 宇都宮大学国際学部研究論集, Vol.49, pp.123−129 (2020)

[78] 棚橋優, 今堀慎治 : "配送計画問題に対するデータベース付きメタ戦略 (最適化の基礎理論と応用) ," 京都大学数理解析研究所講究録, Vol.1879, pp.164–179 (2014)

[79] トランコム株式会社 :「2023 年 3 月期第 2 四半期決算説明会」(2022), https://www.trancom.co.jp/files/topics/1159_ext_02_0.pdf（23/2/7 閲覧）

[80] 菊池亮, 五十嵐大 : "秘密計算の発展 –データを隠しつつ計算する仕組みとその発展–," 電子情報通信学会 基礎・境界ソサイエティ *Fundamentals Review*, Vol.12, No.1, pp.12–20 (2018)

[81] 矢川嵩, 照屋唯紀, 須崎有康, 阿部洋丈 : "Intel SGX における 2 つのリモートアテステーションの利点と欠点の考察," 暗号と情報セキュリティシンポジウム予稿集, No. 1C2-3, pp.1–8 (2022)

[82] 加藤風芽, 新城靖 : "Intel SGX を利用した自己破壊・出現データの実装," コンピュータシステム・シンポジウム予稿集, pp. 42–51 (2020)

[83] Intel : "Code Sample : Intel® Software Guard Extensions Remote Attestation End-to-End Example," (2018) https://www.intel.com/content/www/us/en/developer/articles/code-sample/software-guard-extensions-remote-attestation-end-to-end-example.html（23/8/7 閲覧）

[84] Shamir, A. : “How to Share a Secret,” *Communications of the ACM*, Vol.22, No.11, pp.612–613 (1979)

[85] Rivest, R., L., Shamir, A., and Adleman, L. : “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,” *MIT LCS TM-82* (1977)

[86] Costan, V. Lebedev, I. and Devadas, S. : “Sanctum : Minimal Hardware Extensions for Strong Software Isolation,” in *Proceedings of 25th USENIX Conference on Security Symposium*, pp. 857–874 (2016)