RMX における送信制限機構

安東　翔† 遠山 元道††

†††慶應義塾大学理工学部情報工学科 〒223–8522 横浜市港北区日吉 3–14–1
E-mail: †andy@db.ics.keio.ac.jp, ††toyama@ics.keio.ac.jp

あらまし ルールベースメール配送システム RMX において，先行研究として導入された送信許可機構では，送信者と配送先との関係性まで考慮した詳細な送信制限が可能であり，配送先指定の記述誤りや第三者による望まれないメール配送を防止することが可能であるが，許可する配送先を指定する許可ルールのみの定義により許可設定を行うため，その複雑化が問題であった．また，RMX では，配送先と同様に配送するメールの本文に埋め込む値を動的に取得することが可能であるが，送信許可機構ではそれを考慮した送信制限を行うことができなかった．本論文では，これらの問題を解決するため，許可ルールと拒否ルールの定義により送信制限を行う機構を提案する．

キーワード RMX，メール，メーリングリスト，セキュリティ

1. はじめに

Rule-based e-Mail eXchange (RMX) システム [1–10] は，管理者が予め設定したルールを基にデータベース問い合わせを行い，得られたメールアドレスの集合に対してメールを配送するメール転送エージェントである．先行研究として導入された送信許可機構 [9,10] は，許可ルールの定義，送信者の判別と適用する許可ルールの設定を行うことで，RMX の特徴を生かし，送信者と配送先との関係性まで考慮した詳細な送信制限を可能とし，これにより配送先指定の記述誤りや第三者からのメール送信による，セキュリティ上の問題となり得る望まれないメール配送を防止することが可能となった．

しかしこの送信許可機構では，許可する配送先を指定する許可ルールのみの定義により許可設定を行うため，詳細な送信制限を行いたい場合にその設定の複雑化が問題であった．また，RMX では，生成ルールというものを定義することで，配送先と同様に配送するメールの本文に埋め込む値をデータベースから動的に取得することが可能であるが，送信許可機構ではそれを考慮した送信制限を行うことは不可能であった．本論文では，これらの問題を解決するため，許可ルールと拒否ルールの二つのルールを定義することにより送信制限を行う，送信制限機構を送信許可機構の代わりになるものとして提案する．従来の機構では送信制限に関して許可するという観点からの設定のみ可能であったが，拒否（不許可）という観点からの設定を可能にすることにより，詳細な送信制限の設定の複雑化を避けることが可能となる．また，生成ルールに対する送信制限も実現することで，RMX を用いた高性能な配送を，ユーザがより安全に利用できるようになる．

以下，本稿の構成を示す．まず，2 章で RMX の概要を説明し，3 章で先行研究として導入された送信許可機構に関して，その概要と問題点を述べる．4 章で本研究で提案する送信制限機構について述べ，5 章でその設定例を示す．そして 6 章で評価について述べ，7 章で結論を述べる．

2. 関連研究

電子メールの普及に伴い急増したスパムメールに対する解決策については，今日までに様々なアプローチの研究がなされている．そのアプローチの一つとして，ネットワークの負荷を軽減するために，メールサーバにおいて，差出人情報をもとにメールのフィルタリングを行うものがある．Zisiadis らの研究では，SMTP に代わるプロトコルとして，SMDP (Simple Mail Delivery Protocol) というプロトコルが提案されている [11]. SMDP では，メールのユーザがアドレスを ID として SMDP サーバに登録される．そして受信者が，初めてメールを送信してくる相手のメールの配送を受諾するか否かを選択する．そうすることにより，その選択に基づき SMDP サーバ内にホワイトリストとブラックリストが作成され，以後それを用いて自動的にメールのフィルタリングが行われる．また，Chang らの研究では，同様にユーザがメールサーバにアドレスを登録し，メール送信時に特別な認証を行うプロトコルを提案している [12]. このプロトコルでは，送信者の MTA の IP アドレスを確認し，それが確認できなかった場合は送信元が信頼できないとして，そのメールはサーバにおいて削除される．

これらの研究に対し，本研究では，データベースの情報を用いてメールを配送するという RMX の特徴を生かし，データベースの情報を用いたメールの配送制限を行う，RMX に特化した機構を提案している．データベース上にアドレスが存在するかという，言わば登録の有無や，データベース上で表現することでホワイトリストやブラックリストに基づいた配送制限も可能であるほか，データベースのあらゆる情報を用いて，送信者と配送先との関係性などを考慮した配送制限も可能である．また，配送を許可している配送先には送信し，そうでない配送先には送信しないという “フィルタリング” ではなく，配送を許可していない配送先を一部でも指定している場合は全ての配送を取り止める “送信の制限” であるという点で，上述の研究とは異なる．

3. RMX

Rule-based e-Mail eXchange (RMX) システムは，電子メールとデータベースを組み合わせたメール転送エージェントである．RMX では下記のようなメールアドレスの記述方法により，メールの配送先を指定する．

<RMX のメール配送先指定>:=
<配送ルール名>{<パラメータ>}@ <サブドメイン>.<ドメイン>

RMX はこのようなアドレスを受け取り，指定された配送ルールとそのパラメータに基づきデータベース問い合わせを行い，実際の配送先アドレスを取得する．そうして得られたアドレスに対して最終的にメール配送が行われる（図 1）．

3.1 配送ルール

配送ルールとは，RMX を利用する際に利用者側の管理者が予め定義する必要のある，配送先を指定するために用いるルールで，配送先アドレスを取得する SQL クエリに関連付けられる．RMX は受け取ったアドレスに記述された配送ルールに対応するクエリに，指定したパラメータを挿入しデータベース問い合わせを行うことで，配送先アドレスの集合を得る．このような配送ルールを用いることで，利用者は簡潔な記述で配送先を指定することが可能である．

3.2 演 算 子

3.2.1 配送ルールに用いる演算子

RMX では，それぞれが特別な意味を持つ三つの演算子 “.”，“+” そして “-” を用いることで，複数の配送ルールを組み合わせることが可能である．“.” は積集合を表す演算子であり，複数の配送ルールによって取得される配送先アドレス集合の積集合を得る．“+” は和集合を表す演算子であり，複数の配送ルールによって取得される配送先アドレス集合の和集合を得る．“-” は差集合を表す演算子であり，配送ルールによって取得される配送先アドレス集合の差集合を取る．これらの演算子により，より詳細な配送先範囲の指定が可能となる．

演算子の評価の優先順位であるが，積集合を表す演算子 “.” が他の二つの演算子よりも優先順位が高く，和集合を取る “+” と差集合を取る “-” の優先順位は等しい．また，優先順位の等しい演算子の評価は左から順に行う．

3.2.2 パラメータに用いる演算子

演算子 “+” をパラメータに用いることで，同一の配送ルールにそれぞれのパラメータを与えた際に取得される配送先アドレス集合の和集合が得られる．

また，“-” と “.” はポリモルフィズムの考え方を導入し，一つのルールに複数のパラメータを指定するために用いる演算子である．配送ルールに対し，パラメータの数によって異なる SQL クエリを定義し関連付けておくことで，これらの演算子を用いて与えられたパラメータの数に応じて適当なクエリが呼び出される．

3.3 生成ルール

生成ルールとは，配送先アドレスではなく，配送メールの本文に埋め込む情報を取得する SQL クエリに関連付けて定義するルールである．アドレス上の記述方法は配送ルールと変わらないが，RMX は受け取ったアドレスに記述された生成ルールに対応するクエリに，指定したパラメータを挿入しデータベース問い合わせを行うことで任意の情報を取得し，それを配送メールの本文に埋め込む．生成ルールの定義の例を以下に示す．

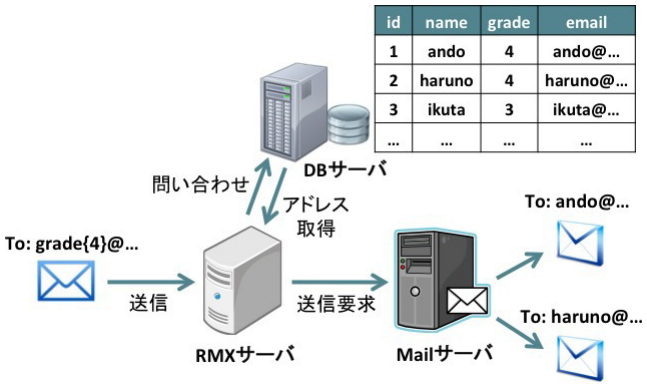

図 1 RMX におけるメール配送の流れ

```
sList = generate
sListType = integer
sList[1] = SELECT s.id, s.name FROM student s
 WHERE s.grade = $1
```

生成ルールは，ルール名に “= generate” を付加することで配送ルールと区別する．上記はパラメータとして与えられた学年の学生の学績番号と名前を取得する，sList ルールである．

3.3.1 生成ルールの組み合わせ

生成ルールを演算子を用いて組み合わせる場合，配送ルール同士を組み合わせる場合とは演算子の表す意味が異なる．生成ルール同士を “.” 演算子を用いて組み合わせることで，各々の生成ルールによって取得される情報を全て，配送メールの本文に埋め込むことが可能である．

また，生成ルールのみをアドレスに用いた場合，生成ルールによって取得された情報が本文に埋め込まれたメールは送信者にのみ配送されることになるが，生成ルールと配送ルールを “.” 演算子を用いて組み合わせた場合，その配送先は組み合わせた配送ルールにより取得されるアドレス集合になる．

4. 送信許可機構

4.1 送信許可機構の導入

RMX では，送信アドレスの記述方法が特殊であるため，送信者が記述を誤り，誤配送を起こしてしまう可能性がある．また，RMX のメールアドレスに対してメールを送信することで，基本的には誰でも RMX を利用したメール配送が可能である．これらは，情報漏洩やスパムメールなどセキュリティ上の問題を引き起こす可能性がある．

そこで RMX に導入されたのが送信許可機構である．送信許可機構の概要を以下に示す．

- 予め許可ルールを SQL クエリを用いて定義
  - SQL クエリを用いて送信者を判別
  - 配送を許可する配送先アドレスを SQL クエリで表現
- 配送先アドレス取得の前に，許可ルールに基づき全ての配送先が許可されているかを判定
  - 全ての配送先が許可されていれば通常通りに配送
  - 不許可の配送先が一つでもあれば，配送を中止し送信者に警告メールを返す

これにより，RMX の特徴を生かし，送信者と配送先との関係性まで考慮した詳細な送信制限が可能となり，望まれないメール配送を防止することが可能となった．

4.2 送信許可機構の問題点

送信許可機構の導入により，送信者ごとの配送先範囲の制限を行うことが可能になった．しかし，その設定は許可ルールのみによるもの，すなわち基本的に配送は不許可とし，許可ルールで表現されているアドレスに対してのみ許可する，というものであるため，配送を許可するグループの一部の人に対する配送を不許可とする場合などに，定義する SQL クエリに集合演算子が含まれるなど，許可ルールが複雑化してしまうという問題点がある．複数の SQL クエリを許可ルールとして定義して，それらを組み合わせる記述方法もあるが，組み合わせる際に使用可能な演算子は和集合演算を表す “+” 演算子と積集合演算を表す “.” 演算子のみであり，差集合演算には対応していないため，この問題を解決することはできない．例えば，図 2 に示すような配送制限を設けることを考えると，左のような配送制限を設ける場合は許可部分を単純な SQL クエリで表現すればよいが，右のような配送制限を設ける場合は，“許可部分 - 不許可部分” を SQL クエリ上で差集合演算子を用いて表現する必要がある．まだこの図の程度の設定であればクエリを定義することはさほど困難ではないが，この入れ子の関係が増えていった場合，クエリは複雑になり，許可ルールの設定とその管理も大変になる．

また，許可の判定は，共に配送先アドレスを取得する配送ルールのクエリと許可ルールのクエリの差集合を取り，その結果の有無により行っているため，現状の送信許可機構では生成ルールに対する配送の制限が実現されていない．

そこで，次章で，これらの問題を解決するため，送信許可機構に代わる送信制限機構の提案とその説明を行う．

5. 送信制限機構

本研究では，従来の送信許可開講に代わるものとして，配送制限の設定や管理がより分かりやすく，生成ルールの制限まで可能とする，高性能な送信制限を行うための送信制限機構を提案する．図 3 に簡単な送信制限の流れを示した．以下の節で従来の機構との変更点を中心に，提案機構について詳しく述べる．

5.1 配送ルールの制限

従来の機構では，送信制限は許可ルールのみによる設定であったため，詳細な配送の制限を行おうとすると定義部分で，差集合演算など集合演算子を含む複雑な SQL クエリを記述する必要があった．

図 2 二つの単純な配送制限の例

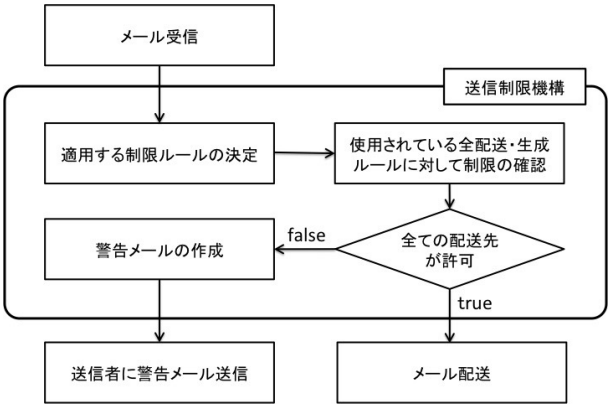

図 3 送信制限機構による送信制限の流れ

そこで提案機構では，許可ルールの他に，配送を不許可とする配送先アドレスを表現する拒否（不許可）ルールを導入する．これにより，許可と不許可の両観点から配送の制限を設定することが可能となり．一つ一つの SQL クエリを単純な形にできるとともに，設定自体が直観的に分かりやすく管理しやすい形となる．ここで，許可ルールと拒否ルールをまとめて制限ルールと呼ぶこととする．

5.1.1 制限ルール

ルール定義の記述方法であるが，拒否ルールの導入に伴い変更を加える．従来の機構では．以下のような記述で，配送を許可する配送先アドレスを指定するための SQL クエリ，送信者の分類を行う SQL クエリであるセレクタ，そして対応するセレクタに該当する送信者に対して許可する，配送先の設定を記述するアプライを設定していた．

Auth[index1] = <配送先指定用 SQL クエリ>
Auth[index2] =
…
Selector[indexA] = <セレクタ>
Apply[indexA] = <アプライ>
Selector[indexB] =
Apply[indexB] =
…

このように許可ルール名を “Auth” で固定し，一つのルールの固まりとしてセレクタとアプライを書き足していく記述方法は，その数が増えれば増えるほど，それぞれがどのような目的で設定されたものであるかを分かりにくくし，変更を加える際などにその管理者への負担も大きくなる．

そこで提案機構では，以下のような記述で設定を行う．

<ルール名> = <allow or deny>
<ルール名>[index1] = <配送先指定用 SQL クエリ>
<ルール名>[index2] =
…
<ルール名>Selector[indexA] = <セレクタ>
<ルール名>Apply[indexA] = <アプライ>
<ルール名>Selector[indexB] =
<ルール名>Apply[indexB] =
…

一つの大きな変更点は，配送ルールや生成ルールと同様に，制限ルールにもルール名を設定者の任意で付けるようにし，別の名前を用いることで複数定義できるようにしたことである．これにより，判別したい送信者が全く異なるセレクタを分離することが可能となり，それぞれに適当な制限ルール名を付けることで，送信制限の管理がより容易になる．複数定義する場合は，“deny1”，“allow2”，“allow3” のように数字を添えて定義することで，その順番にセレクタによる送信者の判別を行い，送信者の該当するセレクタを持つ最初の制限ルールによって判定する．送信者が全ての制限ルールのセレクタに該当しなかった場合，その送信の制限判定は，最後に判別を行ったルールの設定の逆，すなわち最後のルールが許可ルールであれば不許可，拒否ルールであれば許可となる．しかしこれでは，セレクタの設定が不十分であると予期せぬ判定が行われてしまう可能性がある．そこで以下のような記述を加えることで，そのような場合の送信制限方法，すなわち全ての制限ルールに該当しなかった送信者の送信を，許可するかあるいは不許可とするかという，デフォルトの設定を行うことが可能である．

LIMIT_DEFAULT = <allow or deny>

また，アプライにおける配送先の組み合わせの記述に対しては，使用可能な演算子に差集合演算を表す “-” 演算子を加えることで，部分的に例外的な制限の設定をしたい場合などに差集合演算を含む SQL クエリを定義する必要をなくし，単純で分かりやすい設定を可能とする．例として，従来の機構で以下のように member テーブルの中から，ban テーブルに載っている人に対してのみ配送を不許可とする設定があったとする．

Auth[member] = (SELECT email FROM member)
EXCEPT (SELECT email FROM ban)
…
Apply[member] = Default:member

これは提案機構の記述方法では以下のように記述できる．

Auth = allow
Auth[member] = SELECT email FROM member
Auth[ban] = SELECT email FROM ban
…
AuthApply[member] = Default:member-ban

この記述方法であれば，例えば他の条件で抽出した人に対する配送の許可設定を行う際に，同様に ban テーブルに載っている人に対する配送のみ不許可としたい場合，それぞれのクエリに “EXCEPT (SELECT email FROM ban)” を追加することなく，apply において “-ban” を加えるだけで済む．これにより記述量を減らせるとともに，直観的に設定している制限の意味もわかりやすく，ban テーブルに関する不許可設定を止めたいといった場合の変更も容易である．

5.1.2 配送先指定用 SQL クエリの定義種別

提案機構では複数の制限ルールを定義可能としているため，配送先を指定するための SQL クエリの定義を制限ルール内に限ると，同一のクエリを複数の制限ルール内で定義する必要が出てくる場合がある．そのような冗長な記述を減らすために，以下のような形で全ての制限ルールで共通して使用可能な SQL クエリの定義を可能とする．

LIMIT[index1] = <配送先指定用 SQL クエリ>
LIMIT[index2] =
…

このように定義されたものと制限ルール内で定義されたものは，言わばグローバル変数とローカル変数のように扱われ，同一のインデックスを用いることも可能である．そのときは，そのインデックスを用いてクエリを定義した制限ルール内で呼び出す場合はそのクエリが，そうでない制限ルール内で呼び出す場合はそのインデックスで定義された共通のクエリが呼び出される．

5.1.3 セレクタとアプライの別表現

任意に付けたルール名を用いて制限ルールを管理する方法により，分かりやすさは向上したが，一つの制限ルールの定義部分の全ての行頭にそのルール名が付いているため，従来と比較して SQL クエリ定義部分とセレクタ，アプライの設定部分との区別が付きにくくなった．そこで，セレクタ，アプライを以下のように表現し，その区別をはっきりとさせることを可能とする．

Selector[<ルール名>:index] = <セレクタ>
Apply[<ルール名>:index] = <アプライ>

5.1.4 制限ルールによる判定方法

許可ルールのみによる制限を行っていた従来の機構では，以下のような差集合演算を用いた SQL クエリの実行結果により，許可，不許可の判定を行っていた．

（判定する配送ルールのクエリ）
EXCEPT
（対応する許可ルールのクエリ）

このクエリにより取得されるアドレスの数が 0 であった場合は許可，そうでない場合は不許可と判定される（図 4）．

しかし，提案機構において，許可ルールが判定に用いられる場合にはこの判定方法を用いることができるが，拒否ルールが適用された場合にはこの判定方法は使用できない．そこで，拒否ルールが適用された場合に用いる，新たな判定方法を導入する．

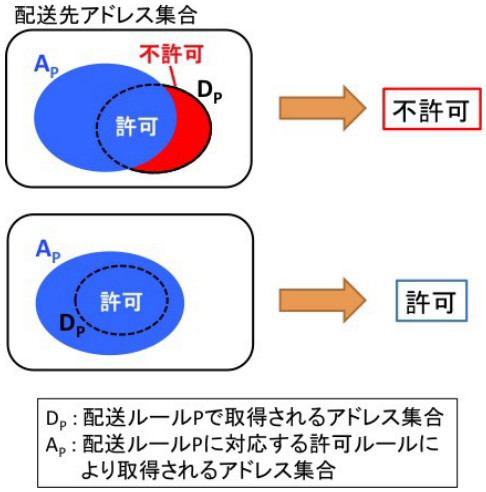

図 4　許可ルールによる判定

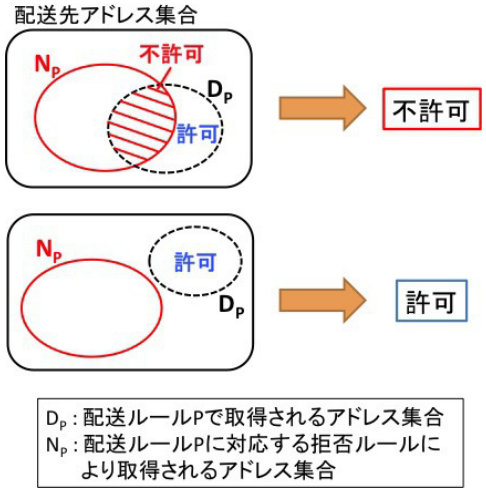

図 5　拒否ルールによる判定

拒否ルールによる許可，不許可の判定には以下の積集合演算を用いた SQL クエリの実行結果を用いる．

（判定する配送ルールのクエリ）
INTERSECT
（対応する拒否ルールのクエリ）

このクエリにより取得されるアドレスの数が 0 であった場合は許可，そうでない場合は不許可と判定する．結果が 0 の場合，配送ルールのクエリで取得されるアドレスのうち，拒否ルールのクエリで取得されるアドレスが一つもないということになる．すなわち，拒否ルールで不許可とされている配送先アドレスは含まれておらず，全ての配送先アドレスが許可されているということである．反対に，結果が 0 出なかった場合は，配送ルールのクエリで取得されるアドレスに拒否ルールで取得されるアドレスが含まれており，不許可とされている配送先アドレスを含むということになる（図 5）．

5.2　生成ルールの制限

従来の機構では実現されていなかった生成ルールの配送の制限を実現するため，提案機構では新たに生成ルール用の制限方法を設ける．

5.2.1　生成ルールの制限方法

生成ルールの制限は，大きく，生成ルールを使用する送信者の制限と，生成ルールを用いた結果生成されるメールの配送先の制限の二つに分かれる．例えば，2 章で例として出した，学生の学籍番号と名前を取得する sList ルールは，professor テーブルに存在する大学の教授と学事担当の職員しか使用が許されず，配送先もその範囲に制限する，といった設定が考えられる．生成ルールはデータベースから情報を取得するというその性質から，配送ルールとは全く異なる制限設定を行うことが考えられるため，生成ルールの制限設定は前述の配送ルールの制限設定とは区別し，それぞれの生成ルールの定義に追加する形で行う．

5.2.2　生成ルールの制限設定の記述方法

生成ルールの制限は，以下のような記述をルール定義に追加することにより設定可能である．

<ルール名>Sender
　＝ <使用を許可する送信者を判別する SQL クエリ>
<ルール名>Recipient
　＝ <配送を許可する配送先を表現する SQL クエリ>

送信者を判別するクエリは制限ルールにおけるセレクタ，配送先を表現するクエリは制限ルールにおける配送先指定用クエリと同様の表現方法で記述する．これらの記述により，送信者判別に該当した送信者がその生成ルールを用いて送信すること，あるいはクエリで表現された配送先がその生成ルールを用いた結果生成されるメールの配送先となることを許可し，それ以外を不許可とするような設定が可能となる．また，生成ルールの送信者と配送先の制限を複数に分けて設定したい場合は，

<ルール名>Sender[<index1>] ＝
<ルール名>Sender[<index2>] ＝
…
<ルール名>Recipient[<index1>] ＝
<ルール名>Recipient[<index2>] ＝
…

という形で，送信者と配送先のそれぞれの設定の中で異なるインデックスを付加して記述すればよい．

5.2.3　生成ルールの判定方法

送信者の制限に関しては，送信者に対して定義した送信者を判別する SQL クエリを実行し，該当した場合は許可，そうでない場合は不許可とする．配送先の制限に関しては，配送ルールの制限と同様に，以下の差集合演算を用いた SQL クエリの実行結果を用いる．

（生成ルールの配送先を指定する配送ルールのクエリ）
EXCEPT
（許可する配送先を表現するクエリ）

このクエリにより取得されるアドレスの数が 0 であった場合は許可，そうでない場合は不許可と判定する．

送信者と配送先それぞれについて制限の設定がない場合は，

全てを許可するものとする．

6. 制限ルールの設定例

この章では，制限ルールの設定例を挙げる．場面は大学を想定し．例に用いるサンプルテーブルを図 6 に示す．配送先は学生に限定し，配送ルールは，name（名前），dept（学科），grade（学年）属性それぞれの値で配送先を決定する，属性名と同名の name，dept，grade ルールを想定する．

このとき

［原則］ 学生（student）と教授（professor）以外の使用は拒否する
［原則］ 学生は送信者本人と同じ学科または同じ学年の学生にメールを送ることができる
［原則］ 教授は送信者本人と同じ学科の学生にメールを送ることができる
［例外］ oda（教授）は全ての学生にメールを送ることができる
［例外］ abe（学生）に対しては全ての学生，教授がメールを送ることができる

これらの送信制限を設けようとした場合，以下のように制限ルールを設定すればよい．

```
LIMIT[all_s] = select email from student
LIMIT[abe] = select email from student where
  name='abe'

basic = allow
basic[student] = select r.email from student r,
  student s where (r.dept=s.dept or
  r.grade=s.grade) and s.email=$sender
basic[professor] = select r.email from
  student r, professor s where r.dept=s.dept
  and s.email=$sender

Selector[basic:student] = from member where
  email=$sender
Apply[basic:student] = Default:student+abe

Selector[basic:professor] = from professor
  where email=$sender
Apply[basic:professor] = Default:professor+abe

Selector[basic:oda] = from professor where
  name='oda' and email=$sender
Apply[basic:oda] = Default:all_s
```

ここに，さらに以下のような送信制限を追加したい場合を考える．

studentテーブル

id	name	dept	grade	email
1	saito	physics	2	saito@...
2	abe	physics	4	abe@...
3	matsuda	chemistry	3	matsuda@...
4	koike	mathematics	4	koike@...
...	...	...	...	...

professorテーブル

id	name	dept	email
1	oda	physics	oda@...
2	yamada	chemistry	yamada@...
3	kato	mathematics	kato@...
...	...	...	...

図 6　サンプルテーブル

［例外］ ban テーブルに載っている学生への配送を禁止する
［例外］ limit テーブルに載っている学生は送信者と同じ学科かつ同じ学年の学生にのみメールを送ることができる

この場合は，以下のように記述を追加，変更すればよい．

```
# 追加
LIMIT[ban] = select email from ban

add = allow1
add[student] = select r.email from student r,
  student s where r.dept=s.dept and
  r.grade=s.grade and s.email=$sender

Selector[add:limit] = from limit where
  email=$sender
Apply[add:limit] = Default:student+abe-ban

# 変更
basic = allow2

Apply[basic:student] = Default:student+abe-ban

Apply[basic:professor] =
  Default:professor+abe-ban

Apply[basic:oda] = Default:all_s-ban
```

7. 評　　価

5 章で例として挙げた提案機構における制限ルールの設定について，従来の機構における許可ルールと比較し，その記述に

表 1　従来の機構と提案機構における制限設定の記述の比較

		提案機構	従来の機構
制限追加前	行数	11	10
	集合演算等を含む SQL の数	0	0
制限追加	追加行数	5	3
	変更行数	4	4
制限追加後	行数	16	13
	集合演算等を含む SQL の数	0	5

表 2　制限の追加による変更行の分類

	提案機構	従来の機構
配送先指定用 SQL	0	3
セレクタ	0	1
アプライ	3	0
その他	1	0

どのような違いがあるか評価する．その比較結果をまとめたものが表 1 であり，設定例において，送信制限を追加する前と後それぞれの，制限ルールの記述行数と集合演算等を含む複雑な SQL クエリの数，そして制限を追加する際に追加，変更が必要な行数を示している．

表 1 から，記述行数は従来の機構の方が少ないことが分かる．これは，提案機構では制限ルールを任意の名前で定義しており，その定義部分で，名前が “Auth” で固定されており拒否ルールの存在しない従来の許可ルールよりも記述量が増えているためである．しかし，行数に関してはそれほど大きな差はなく，むしろ重要なのは集合演算等を含む SQL の数である．提案機構では制限の追加前後で 0 のまま変わらないが，従来の機構では 0 から 5 に増えている．従来の機構では，アプライにおける差集合演算が実現されていないため，配送先の差集合を取りたい場合に，該当するクエリを一つ一つ差集合演算の形に変更する必要がある．また，セレクタの分離，すなわちルールの分離によるセレクタの段階的な適用ができないため，例のように，ある送信者集合の一部に許可範囲を狭める設定を行う場合に，セレクタに関しても，差集合演算や副問い合わせを用いるような複雑な SQL クエリを記述する必要がある．これらが，従来の機構で複雑な SQL が増えてしまう要因である．提案機構では，新しく配送先指定用 SQL クエリを定義し，それを用いてアプライで差集合演算を表現することや，制限ルールを分けることで許可範囲を狭めたい送信者のフィルタをかけることが可能であるため，SQL クエリ自体が複雑になることを避けられる．

表 2 には，提案機構と従来の機構それぞれについて，制限の追加の際に変更が必要となる行を分類した結果を示した．この表からも，従来の機構では SQL クエリに変更を加えているのに対し，提案機構では既に定義した SQL クエリには変更を加えていないことが見て取れる．

8. おわりに

本論文では，RMX おいて，許可と不許可の両観点から配送範囲の制限を設定することが可能である送信制限機構を提案した．制限ルールとして従来の許可ルールに加え拒否ルールを導入し，複数の制限ルールの定義と，配送先アドレスを表現する SQL クエリの組み合わせの記述で差集合を表す演算子 “-” の使用を可能としたことで，クエリの複雑化を防ぎ，直観的に分かりやすくより管理の容易な設定が可能となった．また，生成ルールに対する送信制限も実現したことで，RMX を用いた高性能な配送を，ユーザがより安全に利用することが可能となった．

今後の課題としては，現在，設定ファイルとして Java のプロパティファイルを用いているが，それにより各ルールの定義の表現方法に制約が出てしまい，記述が複雑になってしまっている部分があるため，設定を支援する GUI ツールの開発と，より優れた設定ファイルや設定方法の検討が必要である．

文　献

[1] 高畑 理, 藤沼 健太郎, 石橋 玲, 遠山 元道. “Magic Mirror Mailing：個人情報データベースを利用する柔軟なメイル配送システム”, 情報処理学会データベースシステム研究報告 Pages:123-128 July 2001

[2] Kim Hanki, Sang-Gyu Shin, Motomichi Toyama. “A Rule-Based Mailing System for an Organization”, International Workshop on INformation Processing over Evolving Networks, June 2006

[3] 原田 哲志, 慎 祥揆, 遠山 元道, “RMX における電子メール送受信範囲管理方式の提案”, DBWS2007

[4] 青山 陽亮, 遠山 元道, “RMX におけるポリモルフィックルールとメール本文編集機能の導入”, DEIM2010

[5] 北囿 達也, 青山 陽亮, 遠山 元道, “RMX における関数形式アドレスおよびデバッグ支援機能の実装”, DEIM2011

[6] 小船井 寛, 青山 陽亮, 北囿 達也, 遠山 元道, “RMX におけるルール軍管理機構・エイリアス機構等の実装”, DEIM2012

[7] 松澤 慧, 北囿 達也, 小船井 寛, 遠山 元道, “RMX における受信者別メール本文生成機能および本文参照型アドレスの実装”, DEIM2013

[8] 松本 洋平, 北 和人, 遠山 元道, “RMX における拡張プラグイン機構の導入及び各種プラグインの開発”, DEIM2014

[9] 安東 翔, 遠山 元道, “RMX におけるルール記述に基づく送信許可機構”, DEIM2015

[10] Hiromu Ando, Motomichi Toyama, “Preventing Spam Email by Delivery Limitation in RMX.”, Proceedings of the 19th International Database Engineering & Applications Symposium(IDEAS’15), ACM, 2015. p. 214-215.

[11] Dimitris Zisiadis, Spyros Kopsidas, Leandros Tassiulas, “Simple Mail Delivery Protocol”, Collaborative, Trusted and Privacy-Aware e/m-Services: 12th IFIP WG 6.11 Conference on e-Business, e-Services, and e-Society, I3E 2013, Athens, Greece, April 25-26, 2013, Proceedings. Springer, 2013. p. 100.

[12] Kai-Jie Chang, Chin-Chen Chang, “An e-mail signature protocol for anti-spam work-in-progress”, Proceedings of the 2nd international conference on Scalable information systems. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007. p. 70.