無線LANにおけるチャネル状態依存スケジューリングの実験評価

間 博人[†] 戸辺義人[††] 徳田英幸[†,†††]

本論文は無線 LAN におけるパケットスケジューリング方式 SoCPS を提案し，様々な状況下での無線パケットスケジューリングの有効性を実際の無線 LAN 上で評価する．これまで提案されたチャネル状態を考慮するパケットスケジューリングの多くは，チャネル状態を微小な時間粒度での 2 つの状態（good または bad）として表現する．細かい粒度でチャネル状態の取得する負荷を軽減しソフトウェアでの実現を可能とするために，無線基地局とエンドホスト間の接続性の強度として Strength of Connection（SoC）の概念を導入する．次にチャネル状態を識別方法と，パケットスケジューリングの適用性について調査し，SoC に基づいたパケットスケジューリング方式である SoCPS を設計する．最終的に様々なトラヒックのタイプや無線の状況下で，SoCPS が無線 LAN 全体の効率を改善するかどうかを調査する．FreeBSD3.4 カーネル上に SoCPS を設計，実装し，8 台の無線 LAN ノードを用いた評価結果から，SoCPS の適用性と無線パケットスケジューリングの限界を示す．結果としてリアルタイム UDP 主体のトラヒックでは，無駄なパケットロスを減らし SoC を考慮したスケジューリングは有効である．TCP と UDP が混在するトラヒックにおいても，有効性が確認できた．これに対し TCP 主体のトラヒックでは，TCP の輻輳制御により FIFO であっても十分な通信効率を持つことを明らかにした．

An Experimental Evaluation of Channel-state Dependent Scheduling for Wireless LANs

Hiroto Aida,[†] Yoshito Tobe[††] and Hideyuki Tokuda[†,†††]

This paper describes a packet scheduling scheme for wireless LANs called SoCPS and also explores the effectiveness of wireless packet scheduling under a variety of conditions over an actual wireless LAN. One aspect of the problem in packet scheduling at the base station is the need for channel state dependency. To eliminate a large overhead associated with acquiring a fine-grain channel state as seen in conventional studies and to establish a software-based enhancement, we introduce the notion of Strength of Connection (SoC) which can be expressed as the long-term strength of the connectivity between the base station and an end host. We examine schemes for identifying the channel state and examine the applicability to packet scheduling. We also design a packet scheduling scheme based on SoC called SoCPS. Finally, we investigate the performance under various conditions of traffic and wireless channel states. We have designed and implemented SoCPS on FreeBSD computers. Our evaluation results obtained with up to eight wireless-LAN nodes have determined the applicability of SoCPS and limitation in wireless packet scheduling for wireless LANs. In the TCP traffic, fairness in a short time granularity can be achieved in SoCPS compared with FIFO. However, the network total throughput in a long time granularity does not have the difference between SoCPS and FIFO for the congestion control of TCP. On the other hand, SoCPS improves the usability on the entire network for real time UDP traffic by decreasing the packet loss due to the link deterioration.

1. はじめに

近年無線ネットワークの需要は急激に増加し，無線リンク上のデータ通信が注目されている．特に無線 LAN は，インターネットを構成する要素として欠かせない状況である．本論文では，大学のキャンパスのように移動中に無線 LAN 利用することも可能であるが，大半は教室や研究室で利用する環境を想定する．また移動にともなう複数セル間のローミングは考慮せ

† 慶應義塾大学大学院政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 東京電機大学工学部情報メディア学科
Department of Information Systems and Multimedia Design, Tokyo Denki University
††† 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio University

ず，単一の基地局が扱うことのできる無線 LAN 環境を対象とする．

無線 LAN の帯域は増加傾向にあるが，バックボーンの有線 LAN の帯域はさらに顕著な増加傾向にある．このため無線 LAN の通信において，無線と有線 LAN の接点はボトルネックとなることが多い．こうしたボトルネック地点での公平パケットスケジューリングを実現するため，いくつかの無線公平キューイング方式が提案されている．これらの基本的な共通点は，無線基地局と無線端末間のチャネル状態を基に無線基地局でパケットスケジューリングを行う点である．

無線公平パケットスケジューリングを行うために，無線基地局は無線端末ごとのチャネル状態を識別する必要がある．この際，いくつかの解決すべき問題が存在する．まずチャネル状態の具体的な識別方法は大きな問題である．無線リンクの品質は時間および場所により著しく変動し予測は難しい[3]．また遅延なしのチャネル状態の識別は不可能である．さらに測定の負荷も考慮すべき点である．このようにチャネル状態の表現方法，識別のための機構は，測定遅延，負荷を考慮に入れて論じるべきである．

これまでに提案された方式の多くは，チャネル状態を good，bad の 2 つに分類しているという特徴を持つ．これは，ハードウェアを変更して微小な時間粒度で制御するには十分である．我々は既存の無線 LAN デバイスを用いて，ソフトウェアの変更だけで無線 LAN スケジューリングの改善を目指す．ソフトウェア上に実装する現実的な方式を実現する際には，制御可能なレベルにあって時間軸上で比較的長めの無線強度，すなわち接続性の強度を基本にすることが望ましい．

また，無線パケットスケジューリングとトラヒックの種類の関係は明らかでない．現在の無線 LAN では TCP トラヒックが一般的に使われる．しかし今後無線 LAN 上で音声（VoIP）を利用したアプリケーションは十分考えうる．以上の問題点から，我々は以下のことを述べる．

- **Strength of connection（SoC）の導入**：微小な時間粒度での制御に適した表現ではなく，ソフトウェアで制御可能な時間粒度での接続性の強度を定義する．
- **SoC パケットスケジューリング（SoCPS）の設計と実装**：SoC を指標として用いる無線パケットスケジューリング方式である SoCPS を設計し，FreeBSD への実装を行う．
- **様々な状況での実験評価**：Wave LAN[12] を用い，トラヒックのタイプと無線状態のパターンを複数用意し，SoCPS が無線 LAN 全体の効率を改善するかどうかを実験評価する．

本論文の構成は，次章で本研究の関連研究について述べ，3 章においてチャネル状態を決定する方法について議論する．4 章では SoCPS の設計と実装について述べ，5 章で実験評価を行う．最後に 6 章で本論文をまとめる．

2. 関連研究

既存の無線パケットスケジューリングに関する研究の多くは，Bharghavan らの論文[7] が分類，解説している．チャネル状態依存型パケットスケジューリング（CSDPS: Channel State Dependent Packet Schduling）[1] は，ホストごとのチャネル状態を考慮してスケジューリングを行う最初のアプローチである．CSDPS はチャネル状態が良い場合のみパケットを送信する．このため CSDPS はチャネル状態の違いにより不公平性が生じる．IWFQ[5]，CIF-Q[8]，LTFS[10] はこの不公平性をチャネル状態が回復した後に補償する方法を提案している．IWFQ は，送信しないパケットのサービスフィニッシュタイムが早くなるという特徴を利用することでチャネル状態が回復したフローを優先する．CIF-Q は WFQ を利用した理想値と実際のサービス量の差を，フローごとに変数 log として記録する．変数 log を参照することで CIF-Q は大きく遅滞したフローを優先して補償する．LTFS は補償のための帯域を前もって用意しておく．補償が必要な際にこの帯域を利用することで，他のフローに干渉しない．これらの方式に加え Havana framework[4] では，補償を含んだチャネル状態依存パケットスケジューリングによりアプリケーションレベルでの効率改善を目的とする．Havana はチャネル状態を予測する channel predicator，劣化したチャネル状態のフローを補償する compensator，アプリケーションレベルでのレート制御をする adapter の 3 つのコンポーネントからなる．

これらの既存研究の前提として，無線基地局は各無線ホスト間のチャネル状態を微小な時間粒度で識別する必要がある．Bharghavan らの提案[5] では無線ホストから無線基地局への短い確認応答パケットを用いることで，微小な時間粒度でのチャネル状態を識別する．しかしながら，ソフトウェアレベルでのチャネル状態の取得を考えた場合，微小な時間粒度でのチャネル状態のモニタリングは，大きな負荷が生じる．既存研究では，チャネル状態識別時の負荷を考慮していない．

また，これら既存研究では TCP トラフィック上での無線パケットスケジューリングの性能評価が行われ

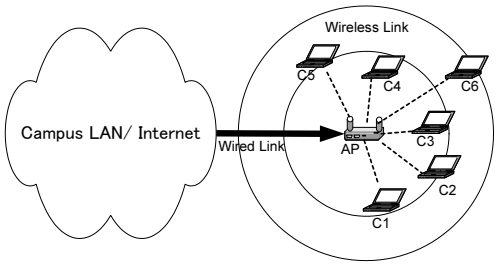

図 1 問題領域
Fig. 1 Problem domain.

ていない．TCP はパケットロスが生じると輻輳制御による自己レート調整機能が働くため，無線パケットスケジューリングが行うレート調整機能とどのような関係にあるかを明らかにする必要がある．また無線 LAN 上の主なトラフィックが TCP であることから無線パケットスケジューリングの有効性を実証するうえで TCP トラフィック時の性能評価は重要である．

本論文は，ソフトウェアが制御可能な時間粒度でのチャネル状態を指標とした無線パケットスケジューリング SoCPS を提案し FreeBSD 上に実装する．また，TCP や VoIP といった実環境を想定したトラヒックにおける無線パケットスケジューリングの性能評価を目標とする．

3. チャネル状態の識別

本章ではチャネル状態の識別法について議論し，本論文で用いる接続性の強度（SoC: Strength of Connection）の定義を行う．SoC は，チャネル状態を値として表現する方法の 1 つである．以後，チャネル状態識別を CSI（Channel-state identification）とする．また S_i は，無線基地局とエンドホスト i 間のチャネル状態を表す．

3.1 問 題 点

図 1 のような 1 台の無線基地局と複数台のホストからなる無線 LAN 環境を想定する．ホストの位置に応じて，パケットの受信できる能力は異なる．たとえばホスト C6 は，C1 よりも基地局 AP との接続性が低い．さらにこの接続性はホストの移動や，マルチパスフェージングや種々のノイズを起因として変動する．こうした環境で S_i を識別する現実的な方法を決定するには，遅延，頻度，粒度の 3 つについて考える必要がある．

遅延：理想的なスケジューリングモデルでは，パケットをスケジュールする時間 t に，すべてのホスト i の S_i をスケジューラが知っていることを想定する．しかしながら，現実的にスケジューラが知ることのできるのは $t - D_i$ ときの S_i である．D_i は i に依存する遅延係数を表す．

頻度：CSI には，チャネル状態の指標として利用できる値のモニタリングが必要である．能動的と受動的手法の双方においてモニタリングには負荷がともなう．さらにチャネル状態が良いときも，定期的にチャネル変動をモニタリングする必要がある．このため高い頻度でのモニタリングは大きな負荷を生じ，帯域は大きく消耗する．このようにモニタリング頻度には，CSI の正確性とコストのトレードオフがある．

粒度：チャネル状態を表す方法はいくつか考えられる．まず good と bad の 2 つの状態としてチャネル状態を表す方法がある．この考え方は，チャネル状態を Markovian モデルとして分析する際に便利である．チャネル状態の他の表現方法として，信号強度，SNR（Signal to Noise Ratio）の値を直接利用する方法がある．SNR はチャネル状態を評価する際の一般的な指標で，受信電力と雑音電力の比から計算する．SNR は次のように定義できる．$\mathrm{SNR} = 10\log_{10}(S/N)$，$S$ は受信電力を表し，N は雑音電力を表す．一般的に変調方式を特定すると，SNR からビット誤り率（BER）P_e の理論値を計算できる．たとえば，P_e に NCR WaveLAN [12] で用いられる PSK（Phase shift keying）を与えた場合を次に示す．$P_e = (1/2)es(S/N)^{1/2}$，es はエラー関数である．このように SNR は BER と密接な関係があり，フローのスループットに影響する．しかしながら SNR は，周辺のノイズやマルチパスフェージングの影響で激しく振動しながら動的に変化することに注意する必要がある．

3.2 Strength of Connection

上述したとおり，S_i の表現方法にはいくつかのデザインがある．CSI の方法を分類する際，粒度と頻度の 2 つの軸がある様子を**図 2** に示す．

図 2 の A は，高い頻度で瞬間的な強度の値を測定する方法に対応する．同様に C は，高い頻度での測定が必要となる方法に対応する．しかしながら，状態は good か bad のどちらかで表す．B は状態を値として表すが，低い頻度である．このため通常，値はある範囲の時間でローパスフィルタを用いる．粒度が two-state のときは瞬間的な状態だけに意味があるため，低い頻度かつ粒度が two-state 時の方法は存在しない．

本研究では，変動する SNR をソフトウェアで制御可能な時間粒度で平均化した SoC（strength of connection）を定義し，チャネル状態の指標として用いる．SoC を使うことで，スケジューリングに関連した

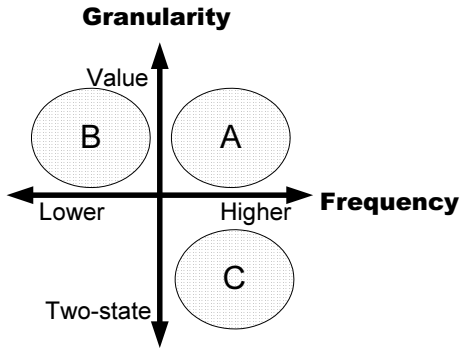

図 2 CSI の区分
Fig. 2 Spectrum in CSI.

通信オーバヘッドを和らげることができる．SoC は SNR と同様 dB で表すが，これは時間で平均した値である．SoC の決定アルゴリズムの詳細は次章で詳述する．

4. SoC パケットスケジューリング

本章は，SoC を用いたパケットスケジューリング方式である SoCPS の設計および実装について説明する．SoCPS は，無線基地局とエンドホストの協調により動作する．次に SoCPS の設計概要および機能の詳細を説明する．

4.1 設計概要

図 3 に SoCPS の構成を示す．SoCPS の制御は，主に CSI と CSI を基にしたスケジューラの 2 つに分かれる．CSI は基地局でそれぞれのホストから受け取った SoC により機能する．スケジューラはこの CSI 情報を使用する．さらに，スケジューリングポリシとしてトラヒックごと，ホストごとかの選択や，他のコンフィグレーションパラメータを与えることができる．このため SoCPS は，様々なスケジューリングポリシを柔軟に実現可能である．

4.2 CSI

それぞれのエンドホストは SNR を定期的に測定する．SNR は無線通信の特性に応じて振動するため，SNR の値は T_{soc} の間隔でローパスフィルタを用いて平滑化する．この際，サンプリングのための時間間隔はフィルタ設計時に問題となる．本論文では，無線基地局と各ホスト間の相互接続性が，秒単位で変動しない環境を想定する．したがってフィルタは，SNR の平均を T_{soc} が 1 秒以下，100 ms 以上の範囲で SoC として計算する．

エンドホストは，T_{soc} の間隔で基地局に SoC を通知する．この SoC は，エンドホストから基地局へのパケットがあれば通常データパケットにのせることができる．

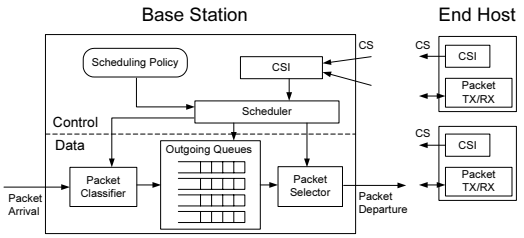

図 3 SoCPS の構成
Fig. 3 Block diagram of SocPS.

```
/* Within T_mon (> T_soc) : */
if (notification of SoC_i comes)
  SoC_i = the notified SoC_i
else /* notification of SoC_i losted
*/
  SoC_i = β × SoC_i /* (β < 1) */
```

図 4 SoC 値の決定方法
Fig. 4 SoC algorithm.

基地局は，受信した SoC 値により soc 情報を更新する．この際，SoC を含んだパケットは連続して失われる可能性がある．この問題を回避するため，我々は**図 4** に示すアルゴリズムを適用し，SoC 値を決定する．β は，減衰係数を示す．

4.3 パケットスケジューリングポリシ

一般的なスケジューリング方式において，フロー間の優先度は weight として表現する．このため，各フローの SoC 値をスケジューリング時に利用する weight へ変換する必要がある．weight 割当ての推移部分を決定するにあたり SoC 値と UDP スループットの関係を調査した☆．この実験結果を基に SoC 値が 0 dB から 10 dB の範囲を weight 割当ての推移部分に決定した．

SoC から weight へ変換するため，Good or Bad（GB）方式と piece-wise liner（PW）方式の 2 タイプを提案する．次に説明のため，C_i をホスト i の割り当てられた通信容量，w_i をホスト i の weight とする．また，$\sum_i C_i = 100$ 単位かつ，w_i の範囲を 0 から 1 までと定義する．たとえば，FIFO 方式では，4 つのノードへ 25 単位が均等に割り当てられる．

GB 方式は，1 つの閾値 D を持つ．もし SoC_i が D を超えた場合 w_i は 1 となる．そうでなければ w_i は 0 となる．**図 5** は，D が 5 dB のときの SoC と GB の weight の関係を示す．また帯域の割当て例を**表 1**

☆ 実験データおよび考察は Appendix を参照．

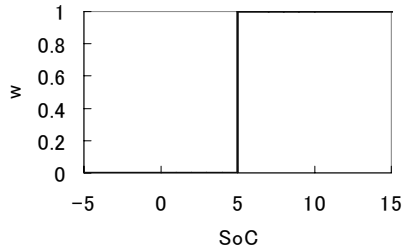

図 5 GB における SoC と weight の関係
Fig. 5 Relationship between SoC and weight in the GB scheme.

表 1 GB 方式での割当て例
Table 1 Example of assignment with GB scheme.

Destination Host	SoC [dB]	w_i	C_i [unit]
Host A	2	0	0
Host B	4	0	0
Host C	6	1	50
Host D	8	1	50

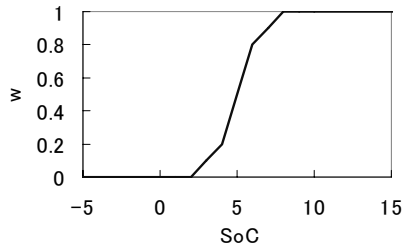

図 6 PW における SoC と weight の関係
Fig. 6 Relationship between SoC and weight in the PW scheme.

表 2 PW 方式での割当て例
Table 2 Example of assignment with PW scheme.

Destination Host	SoC [dB]	w_i	C_i [unit]
Host A	1	0	0
Host B	3	0.1	10
Host C	5	0.5	40
Host D	9	1	50

に示す.

PW 方式は，複数の閾値を持つ．閾値を D_j（$j = 1, \ldots, n$），$D_j < D_k$（$j < k$）と定義する．PW の w_i は，$[D_j, D_k]$ ごとに傾き m_j で線形に増加する．この方式は，GB 方式よりも柔軟な閾値の決定が可能である．**図 6** は，SoC と weight が $D_1 = 2\,\mathrm{dB}$，$D_2 = 4\,\mathrm{dB}$，$D_3 = 6\,\mathrm{dB}$，$D_4 = 8\,\mathrm{dB}$ の場合の関係を示す．また**表 2** に帯域割当て例を示す.

本論文では，ホストごとに帯域を割り当てることを想定している．しかし同様のアルゴリズムを適用することで，トラフィックの種類で分類した帯域の割当ても可能である．SoCPS では WFQ や H-FSC[11] 等の，

```
SoCPS Notations:
  i              : session number
  F(i)           : Packet virtual finish time in session i.
  SF(i)          : Session virtual finish time in session i.
  active_r       : Sum of all active session weight.
  weight         : Weight of the session.
  L              : Length of the packet.
  s_w(s,flag)    : Apply flag scheme and return weight.
  flag           : Scheduling policy. flag{GB, PW etc.}
  soc(dest)      : Search newest SoC of destination.
  soc_prev(dest): Search previous SoC of destination.
Enqueuing:
 1  enqueue(packet P, i)
 2      if not active(i)
 3         activate(i)
 4         active_r += r(i)
 5      if queue(i) is empty
 6         F(i) = SF(i) = max(F(i), V(t)) + L / weight
 7      else
 8        SF(i) += L / weight
 9      put(P, queue(i))
Dequeuing:
 1  dequeue()
 2      i = min(active queues F(i))
 3      P = get(queue(i))
 4      t += L / r
 5      if active(i)
 6         F(i) += Lnext / r(i)
 7      for ever
 8         j = min(active queues SF(j))
 9         tmp_t = prev_t + (SF(j) - V(t)) * active_r / r
10    if tmp_t > t
11            V(t) += (t - prev_t) * r / active_r
12            prev_t = t
13            return P
14         V(t) = SF(j)
15        deactivate(j)
16        active_r -= r(j)
Calculate queue weight:
 1  r(i)
 2      s = soc(destination of session i);
 3      s = (a * s) + ((1 - a)(s_prev(destination of i)))
 4         weight = s_w(s, flag)
 5         return (weight)
```

図 7 SoCPS の疑似コード
Fig. 7 Pseudo-code for SoCPS.

優先度キューイング方式を利用し帯域を割り当てる．H-FSC を優先度キューイング方式として用いた場合の SoCPS の疑似コードを**図 7** に示す．r(i) で各々のキューの重みを計算し dequeue 時に動的に反映する．

4.4 FreeBSD/ALTQ への実装

SoCPS は，FreeBSD 3.4-RELEASE および WaveLAN ドライバに実装した．また，無線基地局のデータパスとして ALTQ[2] を利用した．SoCPS の本体とは別に，以下の拡張を行った．まず，WaveLAN ドライバ “wlp” にブリッジ機能を追加した．`ether_input` 関数の前に宛先 MAC アドレスが local でない場合あるインタフェースから他のインタフェースへ送るコードを追加した．また，ALTQ に対する “wlp” サポートを追加した．ALTQ が提供する H-FSC を含むスケジューリングポリシを利用可能となった．

4.4.1 SoC Frame Sender

エンドホストのモジュールである SoC Frame Sender は，3 つの機能からなる．まず，SoC Frame Sender は，基地局から通知された SoC を算出する．WaveLAN カードはレジスタ上に SNR を持つ．このレジスタから SNR を読むことで，SoC Frame sender は SoC を算出する．次に，SoC Frame Sender は SoC フレームを作成する．このフレームは，IP レベルの

```
struct ether_soc {
  u_long   soc_spa;   /* sender protocol address */
  u_int32_t   soc;   /* SoC */
}

struct socentry {
  u_long   soc_spa;
  u_int32_t   soc;
  struct snrentry   *st_parent;
}
```

図 8 ether_soc と socentry 構造体
Fig. 8 ether_soc and socentry structures.

```
case ETHERTYPE_SOC:
schednetisr(NETISR_SOC);
inq = &socintrq;
break;
```

図 9 ETHERTYPE_SOC の分岐
Fig. 9 Separation of ETHERTYPE_SOC.

情報を参照できないためホストの IP アドレスを含む．この IP アドレスは ALTQ がホストの識別に利用する．**図 8** は，SoC フレームを定義する ether_soc 構造体を示す．最後に SoC Frame Sender は SoC フレームを T_{soc} 間隔で定期的に無線基地局へ送信する．ここで，T_{soc} は 200 ms に設定する．他の T_{soc} 値は，次章において実験する．

4.4.2 SoC Frame Receiver

SoC Frame Receiver は無線基地局で実装され，SoC フレームをエンドホストから受信する．SoC フレームを他の Ethernet フレームと区別するため，新しい ether_type ETHERTYPE_SOC を作成した．さらに */sys/net/if_ethersubr.c* に**図 9** に示す case 文を追加した．

SoC Frame Receiver は SoC フレームを受信後にホストごとの SoC 値を管理する SoC Table を更新する．まず，SoC Frame はエンドホストの IP アドレスを得る．次に SoC Frame Receiver は，このアドレスが図 8 に示す socentry 構造体のエントリに存在するか確認する．もしエントリがすでに構造体に存在すれば，SoC Frame Receiver は socentry 構造体の受信した SoC フレームに基づき SoC を更新する．存在しなければ，socentry 構造体に新しいエントリを作成する．

5. 評　　価

本章では，実験およびその結果について述べる．はじめに実験環境および方法を説明する．次にスケジューリングのオーバヘッドおよび様々な状況下での SoCPS の性能について詳述する．本章の最後で，SoCPS と既存研究との比較考察を行う．

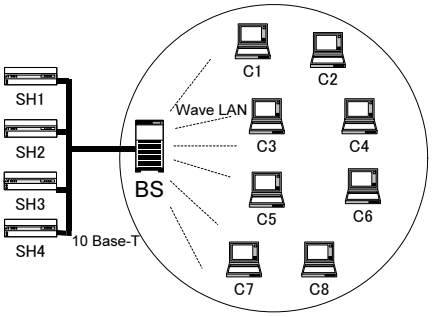

図 10 実験システム
Fig. 10 System for experiments.

5.1 実 験 方 法

SoCPS を FreeBSD 3.4 のカーネル上に実装し，**図 10** に示す実験環境を構築した．SHj は，トラヒックを生成するサーバホスト j，BS は無線基地局，Ci はクライアントホスト i を示す．これらすべてのマシンは FreeBSD 3.4 カーネルを用い，BS と Ci は 2-Mbps の WaveLAN カードを持つ☆．性能は，コンピュータのペンティアムカウンタを使用して測定する．測定時，コンピュータ間の時間の同期を確保するため，すべてのマシンで NTP（Network Time Protocol）を使用する．T_{soc} の値は，最後の移動に関する実験で T_{soc} を減らした効果を明らかにするため，20 ms にした場合を除き 200 ms に設定する．SoC 通知の欠損に対応する減少係数 β は 0.7 とした．

トラヒックを生成させるため，4 台のサーバホストを使用した．それぞれのホストは，2 本の TCP フローか 2 本の VoIP/UDP フローを生成する．VoIP のサイズは UDP/RTP ヘッダを含み 350 バイトである．VoIP ストリームのためのパケットは，30 ms ごとに 93 kbps で送信される．また TCP と UDP パケットは，最大で Ethernet のフレーム長 1500 バイトに収まるように設定される．同じ転送率でのフレームサイズの違いは，基地局でのインタラプト数の違いやオーバヘッドの違いの原因となることに注意する必要がある．

様々なトラヒック状態における性能を評価するため，以下の 4 タイプのトラヒックパターンを考える．

- Tr_{tcp}：各 C_i が TCP フローを受信．
- Tr_{voip}：各 C_i が VoIP/UDP フローを受信．
- Tr_{cbr}：各 C_i が CBR(Constant bit rate)/UDP フローを受信．
- Tr_{tv}：各 C_i（$1 \leq i \leq 4$）が VoIP フローを受信し，各 C_i（$5 \leq i \leq 8$）が TCP フローを受信．

☆ FreeBSD 3.4 の TCP プロトコルスタックは，Reno である．

表 3 環境条件
Table 3 Environment conditions.

Env_g	全ホストの SoC が 15 dB 以上.
Env_b	C_1 の SoC を 10 dB 以下，それ以外の全ホストの SoC を 15 dB 以上.
Env_m	C_1 は移動する．それ以外の全ホストの SoC を 15 dB 以上.

このうち，Tr_{cbr} は，各ホストに 1.5 Mbps の Constant bit rate の UDP トラヒックを送信する．Tr_{cbr} は，VoIP/UDP フローを用いると送信トラヒックが無線 LAN の転送可能容量を下回ってしまう場合に用いる．

これらのトラヒックパターンに加えて，SoC に関連する環境条件を**表 3** に示す．Env_g は，すべてのホストのチャネル状態が良好な環境を想定する．Env_b は，1 台のホストのチャネル状態が悪い環境を想定する．実環境で複数台のホストのチャネル状態が悪いときは，この Env_b における結果がより顕著となることが予測できる．Env_m は 1 台のホストが移動し，他のホストのチャネル状態が良好な環境を想定する．この環境条件では，移動によるチャネル状態の動的な変動への適応性を評価する．

スケジューリングポリシーによる違いを観察するため，FIFO，SoCPS-GB，SoCPS-PW について比較する．SoCPS-GB は，閾値の違いにより 3 タイプを用意する．後の説明において，SoCPS-GB1，SoCPS-GB2，SoCPS-GB3 はそれぞれ 12 dB，6 dB，1 dB の閾値 D を持つ．これは大きい閾値は，弱い接続に対する割当てをより減らすことを意味する．PW では，次の設定を使う．$D_1 = 0$，$D_2 = 4\,\mathrm{dB}$，$D_3 = 10\,\mathrm{dB}$，$D_4 = 14\,\mathrm{dB}$，$w(D_1) = 0$，$w(D_2) = 0.2$，$w(D_3) = 0.8$，$w(D_4) = 1.0$.

5.2 パケット送信のオーバヘッド

はじめにスケジューリングによるオーバヘッドを測定する．BS において CSI スケジューリングをした場合としなかった場合の VoIP/UDP パケットの転送に消費した時間を比較する．実験を 100 回行った後，測定した送信時間の平均と最大値を得た．この送信時間とは受信インタラプトから，送信完了インタラプトまでの時間である．結果を**表 4** に示す．表を見ると SoCPS のスケジューリング方式間には，ほとんど差がない．FIFO と PW を比較すると，0.4 % 平均値が上昇している．このことから SoCPS の負荷は現実的な範囲であるといえる．

次節で全体のスケジューリングのオーバヘッドを観察するため，パケットが損失する環境下で FIFO と SoCPS の全体のスループットを比較する．

表 4 転送時間 [μs]
Table 4 Forwarding time [μs].

Scheduling	Average	Maximum
FIFO	2629	2655
GB1	2638	2668
GB2	2638	2667
GB3	2638	2667
PW	2637	2679

5.3 性能比較

無線通信の評価において，無線状態を再現可能にすることは重要である．再現方法の 1 つに通信状態をトレースしエミュレートする方法がある[9]．この方法は，移動するユーザのシナリオをテストする際に効率的である．対照的に我々は，各ノードは移動可能だが，ほとんど 1 つの無線 LAN のセル内に固定している環境を想定する．そのため本実験は，十分に長い期間実際の測定を行い，短期的な統計効果を抑制する．

実験におけるスループットは，それぞれのホストのユーザレベルのソケット上で測定する．ホストは NTP によってのみ同期しているので，測定した時間は数百ミリ秒の精度しかないと予測される．したがって，スループットの測定は秒単位での算出が望ましい．この実験ではそれぞれのホストは 3 秒ごとにスループットを計算する．Th_i は，クライアントホスト i の 3 秒間のスループットを表す．

5.3.1 TCP 主体のトラヒックの場合

はじめに 8 台すべてのホストのチャネル状態が良好な場合について調べる．良好な状態とは，つねに SoC が 15 dB 以上の場合を意味する．反対に，SoC がつねに 10 dB 以下の場合を不良な状態とする．実際には，無線の特性によりいくらかの乱れが生じる．すべての実験のスナップショットは，C_i（$2 \leq i \leq 8$）を点で記し，C_1 を実線で表す．直観的に TCP フローのスループットの推移は Th_i の帯域がフロー間で平等に割り当てられるためスケジューリング方式に依存しないと予測できる．しかし，**図 11** を見ると 10 秒の粒度で Th_i が振動していることが分かる．SoCPS ではこの振動は起きないことを**図 12** に示す．この違いにかかわらず，**図 13** に示す 300 秒間の全体のスループットで見ると，それぞれのフロー間で均等に割り当てられ，Env_g において FIFO と SoCPS 間のスループッ

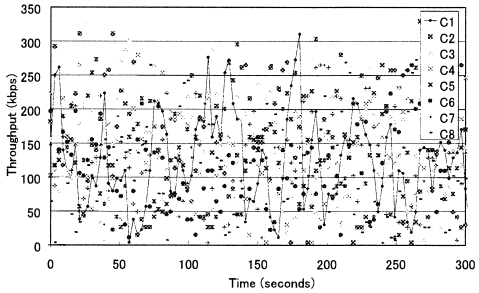

図 11 Env_g-Tr_{tcp}: FIFO
Fig. 11 Env_g-Tr_{tcp}: FIFO.

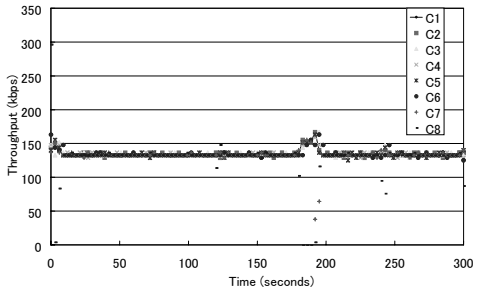

図 12 Env_g-Tr_{tcp}: SoCPS-GB1
Fig. 12 Env_g-Tr_{tcp}: SoCPS-GB1.

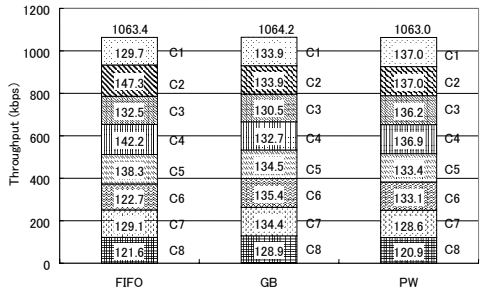

図 13 Env_g1-Tr_{tcp}：各ホストの合計スループット
Fig. 13 Env_g1-Tr_{tcp}: Total throughput.

トの合計に差は見られない．

Env_b で同様の実験を行う．**図 14** は全ホストのスループットの合計である．SoCPS-GB は，閾値 D の選択に応じて，フローへ割り当てられる weight は大きく異なる．SoCPS-GB1 の閾値 D は 12 dB と高い値のため，C_1 の weight は 0 にとどまる．不良なチャネル状態の C_1 への送信を抑制しパケットロスの確率を減らし，結果として SoCPS-GB2，SoCPS-GB3，SoCPS-PW よりも全体のスループットは高い．しかしながら，SoCPS-GB1 全体のスループットはほとんど FIFO と同じである．これは次のことを示唆する．FIFO での，C_1 への送信は，送信側の TCP の輻輳制御により減少する．SoCPS のオーバヘッドは小さ

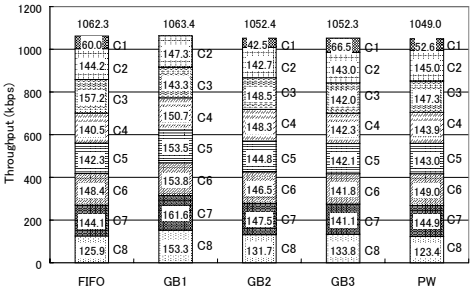

図 14 Env_b-Tr_{tcp}：各ホストの合計スループット
Fig. 14 Env_b-Tr_{tcp}: Total throughput.

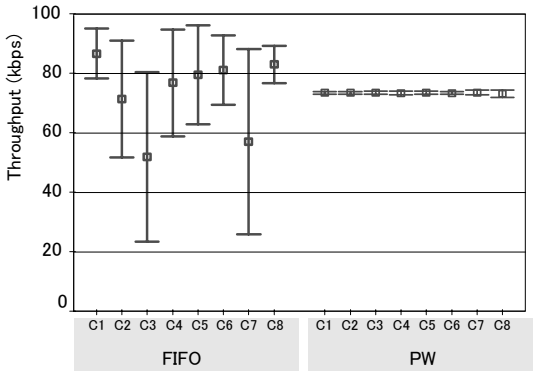

図 15 Env_g-Tr_{voip}：各フローの平均スループット
Fig. 15 Env_g-Tr_{voip}: Average throughput of each flow.

いが，TCP の輻輳制御の性能を超えることは難しい．

結果として TCP 主体のトラヒックかつ，大きな時間粒度での性能は FIFO で十分である．SoCPS は，小さな時間粒度での帯域の均一化が要求された場合に有効である．

5.3.2 連続 UDP フロー主体のトラヒックの場合

次に，UDP フロー主体のトラヒックの場合について調査する．はじめに，8 台すべてのクライアントホストが良好な状態で VoIP/UDP パケットを受信している場合を調べる．**図 15** は，このケースにおける各フローの平均スループットを示す．また，このグラフのエラーバーは標準偏差を表す．FIFO と PW を比較すると FIFO は明らかに各ノードの測定値にばらつきがある．また 300 秒間で平均しても，各ノード間で均等に割り当てられていない．これに対し，PW では均等に帯域が割り当てられている．またほとんど測定値にばらつきがない．結果として TCP の場合と同様に FIFO では，大きな時間粒度でのフローの振動が生じる．VoIP のようなリアルタイムトラヒックにとって，このような振舞いは好ましくない．もしパケットが長期にわたり損失すると，前方誤り訂正（FEC）を用いてもデータを回復できない．このことから，UDP 主

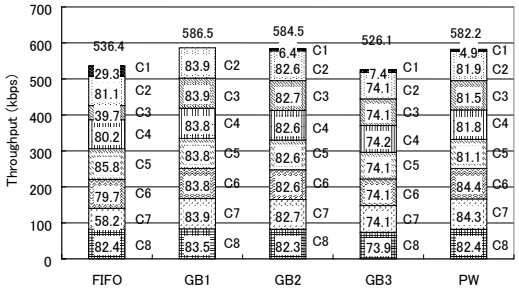

図 16 Env_b-Tr_{voip}：各ホストの合計スループット
Fig. 16 Env_b-Tr_{voip}: Total throughput.

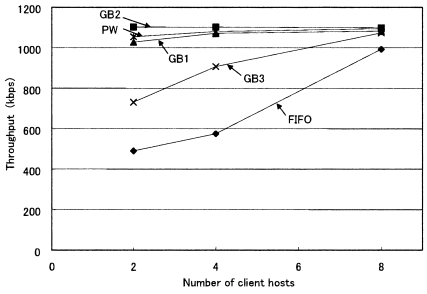

図 17 Env_b-Tr_{cbr}：合計スループットとホスト数の関係
Fig. 17 Env_b-Tr_{cbr}: Total throughput vs. number of client hosts.

体のトラフィックにおいては全ホストのチャネル状態が良好であっても SoCPS は有効である．

Env_b での挙動について詳しく述べる．VoIP フローは TCP フローと違い再送がなくパケットがロスするため，直接スループットの低下につながる．**図 16** は，このケースの合計スループットを示す．このケースでの Th_i は，GB の閾値 D の選択によって結果が大きく異なる．SoCPS-PW はより柔軟に弱い接続性を持つホストへ weight を割り当てる．すべて TCP フローの Env_b とは反対に，C_1 へのフローの weight を減らすことで，全体の通信効率の改善に寄与している．SoCPS-GB3 は閾値を 1 dB と低い値にしているため，ほとんど FIFO の性能と変わらない．クライアントホストの数を変えることで全体の通信効率の改善がより明確になることを**図 17** に示す．

結果として，UDP フロー主体のトラヒックの場合，SoCPS は，低い SoC 値を持つホストへのトラヒックを減らすことで全体のスループットを改善し，効率的である．また，特定の閾値を 1 つ設定することは難しい．複数の閾値を設定可能な SoCPS-PW は SoCPS-GB よりも様々な状況で柔軟に対応可能である．

5.3.3 TCP と UDP が混在するトラヒックの場合

次に我々は，TCP と UDP のフローが混在している環境における実験を行う．C_1 から C_4 までのノードに対し VoIP フローを送信し，C_5 から C_8 までのノードに TCP フローを送信する．このとき C_1 のみ SoC が 10 dB 以下で，他の C_i の SoC は 15 dB 以上である．

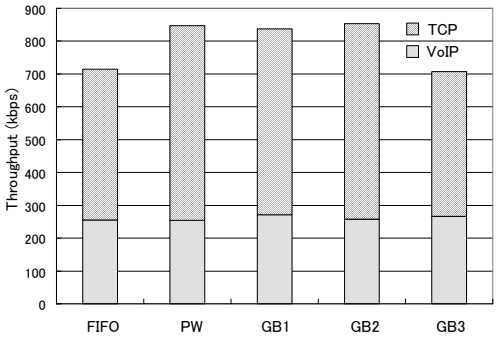

図 18 Env_b-Tr_{tv}: TCP フローと UDP フローの合計スループット
Fig. 18 Env_b-Tr_{tv}: Total throughput.

図 18 はこのケースにおける各方式の合計スループットである．TCP のバーは C_1 から C_4 のホストが受信したスループットの合計を表す．同様に VoIP のバーは C_5 から C_8 のホストのスループットの合計である．VoIP のバーはどの方式もほとんど同じスループットである．これに対し TCP のバーを見ると PW，GB1，GB2 は FIFO と比べスループットが改善されている．これは UDP 主体のトラフィックの場合と同様に C_1 が受信できない帯域を SoCPS では TCP に割り当てることでトータルスループットを改善しているためである．

5.3.4 ホストが移動する場合

最後の実験として，ホストが移動する環境での TCP フローについて調べる．前述したとおり，本論文では大学キャンパスでの無線 LAN 環境を想定している．このため C_1 を持ち運び，部屋の周りを歩いて移動することでホストの移動を実現した．C_1 は，およそ 2 分で部屋の周りを一周し元の場所に戻る移動パターンを繰り返す．**図 19** に C_1 の移動にともなうチャネル状態の変化を示す．図の縦軸は，C_1 で 3 秒おきに算出した SoC 値であり，横軸は実験時間を示す．C_1 は約 70 秒，190 秒時点で BS に接近していることが分かる．

C_1 の SoC 値が，図 19 に示す変化をする環境において PW を適用した場合の TCP スループットの推移を**図 20** に示す．図 20 と，図 19 を比較することで C_1 の SoC 値の変化に帯域の割当てが対応している様子が分かる．

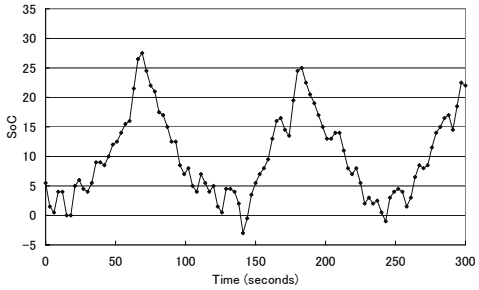

図 19 Env_m: C_1 の SoC 値の変化
Fig. 19 Env_m: SoC fluctuation of C_1.

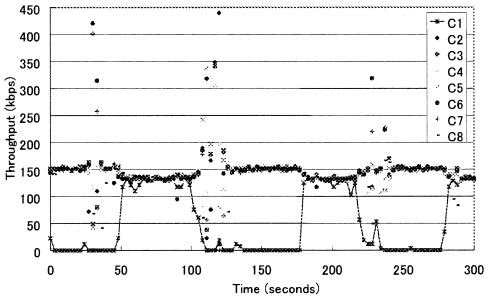

図 20 Env_m-Tr_{tcp}: SoCPS-PW ($T_{soc} = 200\,\mathrm{ms}$)
Fig. 20 Env_m-Tr_{tcp}: SoCPS-PW ($T_{soc} = 200\,\mathrm{ms}$).

また CSI の粒度を短縮した効果を見るために，T_{soc} が 20 ms での実験を行った．**図 21** は，Env_m における合計スループットの比較である．グラフのエラーバーは標準偏差を表す．FIFO と SoCPS を比較すると FIFO は測定値のばらつきが大きい．また T_{soc} が 200 ms のときと 20 ms のときで比較すると 20 ms はすべてのスキームにおいて性能が劣化している．これは，SoC フレーム送受信による帯域の消費と CSI 計算のオーバヘッドのためと考えられる．

T_{soc} の値を小さくすれば，ホストの移動に対しより精度の高いチャネル状態を得ることが可能であるが，オーバヘッドにより性能の低下を招く．逆に T_{soc} の値が大きすぎると，ホストの移動によるチャネル状態の変化に無線パケットスケジューリングのレート制御が対応できず非効率である．このことから実世界で無線パケットスケジューリングを適用する際は，適用する環境のホストの移動速度を十分に考慮し，最適な T_{soc} 値を見つけることが重要となる．本論文が想定するような大学キャンパス内の無線 LAN では人が歩く速度である 1〜3 m/s での移動が中心である．無線 LAN の伝搬範囲が 200 m 程度であることを考えると 200 ms の T_{soc} は，制御する間隔として十分短いといえる．

今回の実験では 1 台のホストが移動する場合の評価を行ったが，各ホストのチャネル状態が頻繁に変化する環境についての考察を加える．ホストが移動することでチャネル状態が悪化や変動が生じる．移動ホストが多いことで，チャネル状態が悪い状況が増え無駄なパケットが増加する．一方，移動ホストが増えても制御パケットや CSI 計算のオーバヘッドは変わらない．SoCPS は無駄なパケットを減らすことができるので，各ホストのチャネル状態が頻繁に変化する環境では改善効果がより顕著に現れる．

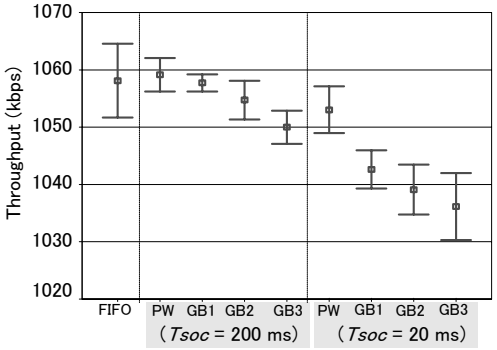

図 21 Env_m-Tr_{tcp}: T_{soc} が 200 ms と 20 ms の合計スループット
Fig. 21 Env_m-Tr_{tcp}: Total throughput of $T_{soc} = 200\,\mathrm{ms}$ and 20 ms.

5.4 既存研究との比較考察

本論文で提案する SoCPS と IWFQ を対象にチャネル状態の識別頻度，補償機能の有無，評価手法の 3 項目に関する比較考察を行う．

チャネル状態の識別頻度

2 章で述べたとおり，IWFQ をはじめとした既存の無線キューイング方式は微小な時間粒度でチャネル状態を識別する必要がある．Tr_{voip} の場合，無線基地局は各無線ホストのチャネル状態をパケット送信間隔の 30 ms よりも細かい粒度で得る必要がある．図 21 においてチャネル状態を取得する頻度を 200 ms から 20 ms に変更することで大きく帯域を消費する様子を示した．

これに対し SoCPS は，大きな時間粒度でのチャネル状態指標として SoC を算出することでチャネル状態識別の負荷を低減できる．またチャネル状態識別の頻度を変えることで，無線ホストの移動速度により必要となる識別精度を得ることも可能である．

補償機能の有無

IWFQ は微小な時間粒度でチャネル状態を識別し，チャネル状態を悪いと判断した無線ホストへのパケットはまったく送信しない．チャネル状態が回復すると，パケットのサービスフィニッシュタイムから今まで送信しなかったパケットを優先して送信する．

この補償機能は瞬間的なチャネル切断に対応可能である．しかし秒単位のチャネル切断の場合，TCP トラヒックはタイムアウトによりパケットを再送する．タイムアウトしたパケットを優先的に送信することは帯域の無駄である．またさらに長期間のチャネル切断に対しては，無線基地局のバッファ容量の制限により補償できない．これらの理由から秒単位でのレート制御を行う SoCPS は，補償機能を持たない．

評価手法

既存研究におけるの評価は，チャネル状態等ステートをすべてスケジューラーが知っていることを前提としている．また，利用するトラヒックも TCP や UDP ではなく一定間隔でパケットを送信した場合のロス率や遅延をシミュレーションを用い評価している．これに対し本研究は，TCP や VoIP といったより現実的なトラヒックを用い評価を行った．

6. まとめと今後の課題

本論文において，実用的な無線パケットスケジューリング方式 SoCPS を提案し，実際の無線 LAN 上における無線パケットスケジューリングの適用性についての実験結果を提示した．SoC を利用することで，現実的な時間粒度での柔軟なスケジューリングを達成した．ソフトウェアに実装可能な形で設計し，FreeBSD 3.4 カーネル上に実装し実験を行うことで，SoCPS の適用性と限界を確認した．

実験結果として 3 点述べる．まず，無線 LAN 上の全体の通信効率を上昇を目的とした場合，UDP 主体のトラヒックの場合 SoCPS は，パケットロスを減らし全体の通信効率は向上する．これらの結果から，トラフィックの種類を管理し，第 2 に UDP と TCP トラヒックが混在している環境でも UDP 主体のトラヒックの場合と同様の SoCPS の有効性を確認した．第 3 として TCP 主体のトラヒックでは，FIFO で十分である．これは，TCP の輻輳制御によりパケットロスを原因とした帯域の占有が少いためである．

これらの結果から，トラフィックの種類を管理しスケジューリングポリシを切り替えることの有効性を今後検討する必要があると考える．また本研究では，ソフトウェアの変更のみで無線のチャネル状態を識別し，これをスケジューリングに反映するため，スケジューリングに大きな時間粒度での接続性の強度を示す SoC を導入した．今後は，SoC を導く際の最も効率的なフィルタの設計についてさらに検討する必要がある．

参考文献

1) Bhagwat, P., Bhattacharya, P., Krishna, A. and Tripathi, S.: Enhancing throughput over wireless LANs using channel state dependent packet scheduling, *Proc. IEEE INFOCOM '96* (Apr. 1996).
2) Cho, K.: A framework for alternate queuing: towards traffic management by PC-UNIX based routers, *Proc. USENIX 1998 Annual Technical Conference* (June 1998).
3) Eckhard, D. and Steenkiste, P.: Measurement and analysis of the error characteristics of an in-building wireless network, *ACM Computer Communication Review*, Vol.26, No.4, pp.243–254 (Oct. 1996).
4) Gomez, J., Campbell, A.T. and Morikawa, H.: The Havana framework for supporting application and channel dependent QoS in wireless networks, *Proc. IEEE ICNP '99* (Nov. 1999).
5) Lu, S., Nandagopal, T. and Bharghavan, V.: Fair scheduling in wireless packet networks, *Proc. ACM SIGCOMM'97*, pp.63–74 (Aug. 1997).
6) Mathisen, T.: Pentium Secrets, *Byte magagize* (Jul. 1994).
7) Nandagopal, T., Lu, S. and Bharghavan, V.: A unified architecture for the design and evaluation of wireless fair queuing algorithms, *Proc. ACM MOBICOM'99*, pp.132–142 (Aug. 1999).
8) Ng, T.S., Stoica, I. and Zhang, H.: Packet fair queuing algorithms for wireless networks with location dependent errors, *Proc. IEEE INFOCOM'98* (Mar. 1998).
9) Noble, B.D., Satyanarayanan, M., Nguyen, G.T. and Katz, R.H.: Trace-based mobile network emulation, *Proc. ACM SIGCOMM'97* (Sep. 1997).
10) Ramanathan, P. and Agrawal, P.: Adapting packet fair queuing algorithms to wireless networks, *Proc. ACM MOBICOM'98* (Oct. 1998).
11) Stoica, I., Zhang, H. and Ng, T.S.E.: A hierarchical fair service curve algorithm for link-sharing, real-time and priority service, *Proc. SIGCOMM'97*, pp.249–262 (Sep. 1997).
12) Tuch, B.: Development of WaveLAN, an ISM band wireless LAN, *AT&T Technical Journal*, Vol.72, No.4, pp.27–37 (Jul./Aug. 1993).

付　　録

A.1 予測されるスループットと SoC の関係

有線ホストから無線基地局を介し，無線ホストへ 1 Mbps の UDP フローを送信する実験を行った．受

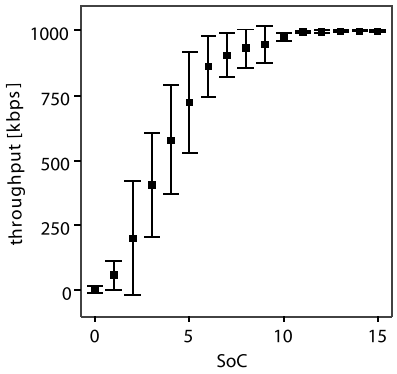

図 22 SoC とスループットの関係
Fig. 22 SoC vs. throughput.

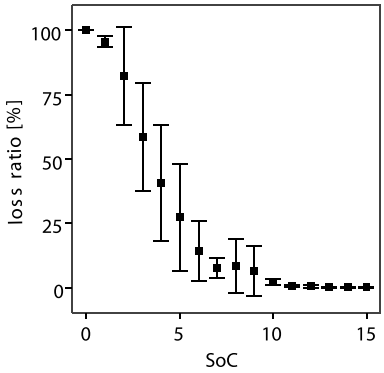

図 23 SoC とロス率の関係
Fig. 23 SoC vs. loss ratio.

信側の無線ホストは，移動しながら SoC 値，スループット，ロス率を 3 秒間隔で 30 分間測定した．**図 22** に無線ホストで計測した SoC 値と受信スループットの関係を示す．また，**図 23** は，SoC 値とロス率の関係である．各グラフの値は平均値，エラーバーは標準偏差を表す．

図 22 を見ると 3 秒の時間粒度で見た場合接続状態，切断状態の 2 通りではなく中間の推移部がある．この推移部では SoC 値が小さくなればスループットが低くなる関係にある．これは，チャネル状態の悪化にともない．パケットロスが生じた結果である．また図 23 をみると，より明確に SoC 値の低下にともないパケットロス率が増加する様子が分かる．

(平成 14 年 3 月 25 日受付)
(平成 14 年 10 月 7 日採録)

間　博人（学生会員）
現在慶應義塾大学大学院政策・メディア研究科修士課程在学中．無線通信，通信プロトコルの研究に従事．

戸辺　義人（正会員）
東京電機大学工学部情報メディア学科助教授．博士（政策・メディア）．ネットワークプロトコル，インターネット計測技術，ユビキタスネットワークの研究に従事．ACM，IEEE，電気学会，電子情報通信学会会員．

徳田　英幸（正会員）
慶應義塾大学より工学修士．カナダ，ウォータールー大学より Ph.D.（Computer Science）．現在，慶應義塾大学大学院政策・メディア研究科委員長．分散リアルタイムシステム，マルチメディアシステム，通信プロトコル，超分散システム，ユビキタスシステム等の研究に従事．IEEE，ACM，日本ソフトウェア科学会各会員．