LL-003

SNMPを用いた汎用プリント枚数取得システム
A SNMP based generic print job accounting system

桝田 秀夫† 齊藤 明紀‡ 中宮 広揮§ 増澤 利光‡
Hideo Masuda Akinori Saitoh Hiroki Nakamiya Toshimitsu Masuzawa

1. まえがき

教育用計算機システムでは，運用コスト削減が大きな課題となっている．その中でも，プリンタシステムに着目したコスト削減に着目する．プリンタシステムの消耗品には、大きく分けてトナーと紙があるが、運用コストの削減には、まず、無駄な印刷を減らし、限られた資源(トナー、紙)を有効に活用させることが重要である。このような問題を解決するため，利用者ごとの印刷可能枚数に上限を持たせるという方法が利用されるようになってきている．

現在，このような要求を満たすシステムはいくつか存在している [1, 2, 3, 4]．RICOH 社の Ridoc IO Gate[3] や FujiXerox 社の DocuHouse[4] といった商用のシステムは，各社の専用のプリンタにしか対応しておらず，導入コストが高くつく傾向にある．文献 [2] は，印刷ジョブごとに，Postscript プリンタと互換性を持つ Ghostscript と呼ばれるインタプリタを用い，あらかじめ枚数を計測して印刷枚数を取得し総印刷枚数を制限するシステムである．この方式では，実際にプリンタで印刷する前にそのジョブの印刷枚数が確認できる為，制限枚数を越えてしまう印刷を破棄できる．しかし，プリンタ自体の Postscript インタプリタとの互換性の不足などにより，プリンタでは印刷できるのに，Ghostscript では処理できずに結果的に印刷ジョブを破棄してしまったり，印刷枚数カウントが食い違ってしまうといった問題が指摘されている．文献 [1] は，印刷ジョブの処理の直前と直後にプリンタの総印刷枚数カウント情報を取得し，その差を印刷ジョブ自体の印刷枚数と見做すシステムであり，本学サイバーメディアセンターの情報教育システムで実際に稼働している．この方式では，実際にプリンタで印刷した枚数が計測できるが，総印刷枚数カウント情報の取得方法が，特定の Postscript プリンタにしか対応しておらず，また，印刷ジョブ間の間隔が空いてしまうことにより，プリンタのスループットが大幅に低下することが確認されている．

そこで本報告では，文献 [1] の問題点を改善するため，総印刷枚数取得システムの改良を行う．

2. システムの概要

本報告では，新たな総印刷枚数取得方式として SNMP を用いた方式に着目する．SNMP はネットワーク機器を管理するプロトコルであり，管理対象の機器の持つ，MIB というデータベースの情報を元に管理を行う．プリンタに関しては，1995 年，PrinterMIB が定義されている [5]．本研究で提案する方式は，この PrinterMIB を参照して得られる情報をもとに，ジョブごとの印刷枚数を算出する．PrinterMIB 自体は RFC により定義された統一的な規格であり，近年，PrinterMIB に対応したプリンタの普及率は問題にならない程度にまで上がってきているので，この方式は，文献 [1] のシステムの汎用性の問題を解決することが期待できる．また，PrinterMIB は従来の方式に比べて，対象となるプリンタに対するより詳細な情報が参照できるので，システムを高効率化する事が期待できる．

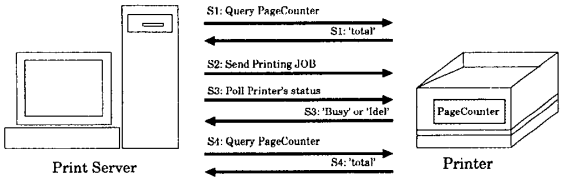

図 1: システム構成とその動き

3. 実装

本システムは，図 1 のように，プリンタサーバとプリンタからなり，ネットワークで接続されている．

3.1 アルゴリズム

印刷ジョブ毎の印刷枚数を取得するアルゴリズムは，以下のようになっている．

Step1 ：ジョブをプリンタに送付する前にプリンタの総印刷枚数カウンタ値 *total* を取得する．

Step2 ：ジョブをプリンタに送付する．

Step3 ：プリンタが印刷を完了したかどうかをポーリングで確認する．

Step4 ：印刷完了後に，総印刷枚数カウンタ $total'$ を取得する．

Step5 ：$total' - total$ を，このジョブの印刷枚数として記録する．

3.2 既存システムの問題点

文献 [1] のシステムでは，プリンタ内部の Postscript エンジンに対して，総印刷枚数カウンタ読みとりコマンドを送ることで，総印刷枚数カウンタ値を取得している．しかし一般に，プリンタは，ラスタライザ (Postscript エンジン) とメディアパス (カウンタを含む紙が通る部分) がパイプライン的に動作しており，Postscript エンジンが印刷完了になったとしても，総印刷枚数カウンタが正しい値になっているとは限らない．そのため，文献 [1] のシステムでは，Step3 と Step4 の部分を，以下のようなアルゴリズムで実現している．

Step3-1 ：1 枚目が出力されるのにかかると予想される時間だけ待つ．

†大阪大学サイバーメディアセンター
‡大阪大学大学院情報科学研究科
§大阪大学基礎工学部情報科学科

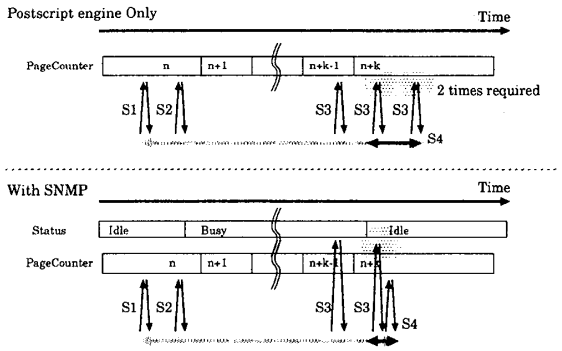

図 2: 総印刷枚数取得までのタイムダイアグラム

Step3-2 : 総印刷枚数カウンタ取得コマンドを送って，カウンタ値を読みとる．

Step3-3 : 1 枚が排出されるのにかかると予想される時間だけ待つ．

Step3-4 : 総印刷枚数カウンタ取得コマンドを再度送って，前回のカウンタ値から変化があった場合は印刷中と判断して，Step3-3 に戻る．

Step4 : 印刷完了として，前回取得したカウンタ値を返す．

文献 [1] のシステムでは，利用しているプリンタ (EPSON 社製 LP8400PS3) の性能から算出し，Step3-1 が 10 秒，Step3-2 が 5 秒となっているため，複数のジョブをプリンタに送りたい場合に，実際に印刷が完了してから，平均 7.5 秒のアイドル時間が入ってしまうことが，大きくスループットを下げてしまっていた (図 2 上)．

3.3 改善方法

本システムでは，Step3 には PrinterMIB のうち，以下のいずれかの値¶を用いる．

- PrtMarkerLifeCount (mib-2.43.10.2.1.4)
- PrtMarkerPowerOnCount (mib-2.43.10.2.1.5)

また，Step4 には PrinterMIB のうち，以下のいずれかの値¶ を用いる．

- PrtConsoleOnTime (mib-2.43.13.17.6.1.2.1)
- PrtMediaPathStatus (mib-2.43.13.4.1.11.1)
- PrtMarkerStatus (mib-2.43.10.2.1.15)

これにより，総印刷枚数カウンタの変化が無くなることを待つといった無駄が削減されるので，スループットの向上が期待できる (図 2 下)．

4. 評価

前節の実装を表 1 の機材を用いて構築し，プリンタ本体の実際の印刷完了から，総印刷枚数を取得するまでの処理時間を評価した．評価方法として，複数の印刷ジョブを送信し，最初の印刷ジョブの一枚目の紙送りが開始されてから，最終印刷ジョブの最後の紙が排紙トレイに落ちるまでを目視によって計測し，本システムが存在しない場合と存在する場合で比較した．

¶プリンタによっていずれかが存在しない場合がある．

プリンタサーバ	RedHat Linux 9, LPRng
プリンタ	FujiXerox DocuPrint C2221 EPSON LP9200PS3
ネットワーク	100baseTX

表 1: 評価に使用した機材

Step3 でのポーリング間隔を 1 秒に設定したところ，平均 0.7 秒程度の処理時間で総印刷枚数の取得が出来ていることが確認できた．ポーリング間隔をさらに縮めることにより，取得に掛かる処理時間が短縮されることが期待できるが，使用したプリンタでは，0.1 秒以下にした場合に，SNMP による問い合わせが実質的に応答しない状態に陥った．これは，SNMP モジュールの CPU が過負荷状態に陥ったものと推測できる．このことから，ポーリング間隔として 1 秒を選ぶことは，SNMP モジュールに対する負荷としても適切であると予想される．

5. まとめ

本報告では，プリンタからの総印刷枚数取得方式として SNMP を用い，汎用性やスループット低下の問題を改善する手法の設計と実装を行った．本方式は，PrinterMIB に対応したプリンタの普及率は上がってきているため，汎用性は高く，実装したシステムを既存のシステムと比較して評価したところ，処理時間が 10 分の 1 に削減できたことが確認できた．

今後の課題として，プリンタ機種毎の PrinterMIB の表現値の差異を吸収する設定ツールの拡充や，このサブシステムを元にして，単なる総印刷枚数制限だけでなく，モノクロとカラーの違いなどのプリンタ種別による差異や，授業時間内の印刷と自習時間での印刷を別の括りにして算定するといった統合管理システムを構築していくことが考えられる．

参考文献

[1] 桝田, 中村, 近藤, 齊藤, 中西 : “教育用計算機システムにおける印刷システムに求められる要求とその実装について”, 火の国情報シンポジウム 2002, pp. 107–114, 2002.

[2] 丸山, 藤井, 中村 : “出力制限の可能な印刷システムの構築と運用”, 平成 13 年度情報処理教育研究集会, p.342, 2001.

[3] リコー社, “Ridoc IO Gate”, http://www.ricoh.co.jp/IPSiO/utility/iogate/.

[4] 富士ゼロックス社, “DocuHouse”, http://www.fujixerox.co.jp/product/soft/docuhouse/.

[5] R.Smith, F.Wright, T.Hastings, S.Zilles and J.Gyllenskog : “PrinterMIB v2”, RFC1759, 1995.

謝辞

本研究は一部，日本学術振興会・科学研究費補助金・基盤研究 B(2)（課題番号 15300017）の研究助成による．