Inverse Multiplexing in Short-Range Multi-Transport Wireless Communications

Lichen Dai
Heidi R. Duffin

See next page for additional authors

Follow this and additional works at: http://scholarsarchive.byu.edu/facpub

Part of the Computer Sciences Commons

Original Publication Citation

BYU ScholarsArchive Citation
http://scholarsarchive.byu.edu/facpub/506

This Peer-Reviewed Article is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in All Faculty Publications by an authorized administrator of BYU ScholarsArchive. For more information, please contact scholarsarchive@byu.edu.
Inverse Multiplexing in Short-Range Multi-Transport Wireless Communications

James C. Funk, Heidi R. Duffin, Lichen Dai, Charles D. Knutson
Computer Science Dept.
Brigham Young University
Provo, UT 84602
{funk,hduffin,lcd,knutson}@cs.byu.edu

Abstract—This paper describes a mechanism for utilizing Inverse Multiplexing to significantly increase the bandwidth available to short-range wireless devices. Previous work with Inverse Multiplexing has focused on wired networks; its implementation with short-range wireless transports introduces heterogeneity in the links, which must be taken into account. A mathematical model for an Inverse Multiplexing system is derived for several scheduling algorithms. Both Process Limited and Transport Limited systems are examined. The validity of this model is shown by our implementation of an Inverse Multiplexing layer that uses IrDA and Bluetooth transports. Concepts related to Inverse Multiplexing such as usage models, negotiation, Quality of Service, and the simultaneous use of multiple Bluetooth transports are discussed.

I. INTRODUCTION

Lack of bandwidth is a significant problem in some mobile devices [1]. Bandwidth may be increased through various means with accompanying tradeoffs. Increasing the width of available frequency utilized by a transport will proportionally increase bandwidth, but it will also increase the complexity of the device and may cause it to violate government RF Spectrum regulations. Impressing compression schemes may also increase effective bandwidth, but may require excessive resources for processing and may significantly increase latency. An alternative to these approaches is to use Inverse Multiplexing (IM), which increases bandwidth by using multiple transports simultaneously. As illustrated in Figure 1, a data packet is segmented and sent across separate transports to a receiver which reassembles the data. While IM may be used in combination with other techniques, the main benefit is that it does not require the development of a new transport, allowing the utilization of current transports with little or no modification.

This paper demonstrates the validity of IM as a quick and inexpensive way to increase the bandwidth available to wireless devices. The benefits of IM are shown both by mathematical analysis and by experimentation with hardware. The remainder of this paper is organized as follows. Section II reviews previous work with IM in various applications. Section III introduces our extension of IM into short-range wireless communications and discusses various scheduling algorithms explored in the paper. Section IV presents our mathematical analysis of IM and predicts the resulting performance of systems using IM. Section V displays our implementation and experimental results. Section VI describes some of the future research areas and discusses concepts related to IM such as usage models, negotiation, Quality of Service (QoS), Quality of Transport (QoT), and the simultaneous use of multiple Bluetooth transports. Section VII presents conclusions.

II. PRIOR WORK

IM has been used in a variety of applications with the details of implementation dependant upon the usage model. In any case, a complete solution must include:

- A means for determining when and where to impose Inverse Multiplexing in normal data flow
- A mechanism for segmenting data
- A mode for scheduling the links
- A fast and accurate reassembly method

The concept of modem Bonding takes advantage of Internet practices to increase transmissions where possible [2][3]. For instance, multiple HTML packets from a single web site may be split between two modem lines. Alternatively, using smart downloading, the first half of a download may be transferred on one modem line and the second half on the other. A valuable idea gained from these approaches is that of using a second modem only as needed when the bandwidth demand increases, such as during a large download.

RFC 1990 [4] defines the Multilink PPP protocol which is used widely in modern ISDN equipment [3]. This protocol implements Inverse Multiplexing at the link layer and focuses on methods of handling multilink negotiation, segment loss, and receiving-side buffers. It suggests the addition of a header to each segment that includes, among other things, a sequence number. The protocol also involves the notion that packets can be segmented and scheduled in two ways. If variable sized segments are used, they should be sized proportional to the link transmission rate. If segments are of identical size, multiple segments should be distributed to the links according to transmission rate.

Researchers have explored methods for intelligently scheduling links to achieve improved transmission rates. Adiseshu et al. [5] utilize a reversed Fair Queuing algorithm to choose the...
next output channel for each packet. An example of this is Surplus Round Robin, where each channel is allotted a number of units of data that can be sent at a time. However, this method does not determine the appropriate unit of data for each channel in the presence of channels with different transmission rates. Snoren [6] attempts to overcome this problem by using a Weighted Round Robin scheduler that continuously adapts in the presence of congestion. Unfortunately, this scheme adds additional time and complexity that may not be suitable to one-hop networks.

Another unique approach to Inverse Multiplexing is Bit-Based [7]. This implementation runs on specialized hardware with up to eight T1 or E1 lines. Using this technique, each bit of data is siphoned to a different link in Round Robin fashion, using buffers at the receiving side to compensate for variable physical link delays. This approach is best suited to an implementation that runs over transports which have nearly identical transmission speeds and stable connections. In a wireless environment where the implementation potentially runs over heterogeneous transports with the inherent instability of wireless connections, a more flexible approach is needed.

III. INVERSE MULTIPLEXING OVER HETEROGENEOUS TRANSPORTS

Implementing Inverse Multiplexing in a short-range wireless environment necessitates an approach different from the typical wired scenario. Short-range wireless devices are typically limited by power consumption constraints and hardware costs. Further, they operate in a frequency domain that is prone to interference. In addition, wireless devices are more likely to be equipped with multiple transports, and those transports typically have significantly different characteristics. Therefore, an Inverse Multiplexing solution for short-range wireless devices should:

- Be simple, to accommodate low hardware cost and low power consumption;
- Handle transport instability and interference;
- Perform well over transports with different characteristics.

Our approach incorporates these concepts by introducing an additional layer in the wireless transport stack called the Inverse Multiplexing Layer (IML). The IML resides above the transport layer, hence it can accept a variable number and type of transport stacks beneath it without the additional complexity required to translate between various protocols.

The IML breaks large packets into uniform segments and adds a sequence number to the header of each segment. By using uniform segments rather than variable ones, IML saves the time and resources necessary to transmit the segment size. The sequence number is used for segment reordering at the receiving side since segments are not guaranteed to arrive in order.

The IML should use an optimal schedule to determine which transport to send each segment across. An optimal schedule is defined to be one that keeps all transports as busy as possible. Such a schedule maximizes transmission speed by sending data across the transports proportional to their transmission rates. This causes more data to be sent on the faster transports.

Various algorithms can be used to approximate an optimal schedule. For homogeneous transports a simple Round Robin (RR) algorithm could be used to provide an optimal schedule since the transport characteristics are identical. For heterogeneous transports, the optimal schedule can be approximated using either a Weighted Round Robin (WRR) or Weighted Random (WR) algorithm. In the WRR scheme, the weights indicate the number of segments that are sent over one transport before moving to the next transport. In the WR scheme, the weights indicate the probability for a particular transport to be scheduled next. These algorithms are optimal only when the weights are properly set; otherwise, using Inverse Multiplexing may degrade performance instead of improving it.

IV. MATHEMATICAL MODEL

Inverse Multiplexing only works when different transports are able to send data in parallel. An accurate model must take into account the portion of the transmission that may be done in parallel and that which must be done serially (see Figure 2). The time needed to send a segment across a transport may be divided into three parts:

1) Sender Time. The time needed by the sending device to process outgoing data for the segment, packetize the data, and schedule transports for the segment. This processing must be done serially because it uses resources (such as CPU, memory and main I/O bus) that are needed for all transmissions. For simplicity we will ignore the cases where these resources are sharable or duplicated. The time required to send depends on the nature of the transport, the size of the segment, the number of transports and the complexity of the scheduling algorithm. We represent this time period as SenderTime(i, s) for transport i and segment of size s.

2) Transport Time. The time needed by the hardware transmitter to transmit the segment. This period includes queuing delay as well as the time spent for physical transmission. For simplicity our model will assume that all transceiver hardware acts independently from other transceivers. We represent this time period as TransportTime(i, s) for transport i and segment of size s.

3) Receiver Time. The time needed by the receiving device to process incoming data from hardware transceivers. Incoming data may only be received serially for the same reasons given for SenderTime(i, s). We represent this time period as ReceiverTime(i, s) for transport i and segment of size s.

Approximate transmission rates for Inverse Multiplexing will now be derived for Round Robin and optimal scheduling, which

---

758
may be approached with WRR and WR scheduling. Our efforts focus on predicting the steady state transmission rate, ignoring latency. For this reason we ignore the processing time needed at the beginning and end of data transmission. Models will be formed for systems that are both Transport Limited and Process Limited. The term Transport Limited refers to a system in which the limiting factor of the transmission rate is the time needed by the transports to transmit the data. The term Process Limited refers to a system in which the limiting factor of transmission rate is the processing speed of either the sending device or the receiving device.

The following notation will be used in the derivations:

- \( P \) is the packet size in bits
- \( S \) is the segment size in bits \(^1\)
- \( N \) is the number of transports
- \( Tr_i \) is the TransmissionTime
- \( T_i \) is the TransportTime
- \( S_i \) is the SenderTime
- \( R_i \) is the ReceiverTime
- \( Pr \) is the Proportionality

A. Round Robin

The limitation of using Round Robin scheduling may best be seen by the following example. Figure 3 illustrates an IML system with two transports. Both transports require the same amount of processing time in the sender and the receiver. Transport 2 requires much more transport time than Transport 1. The transport time for Transport 2 is also greater than the process time needed by the sender or receiver to process segments for both transports. Hence, Transport 2 is clearly the limiting factor.

With Round Robin scheduling, half of the segments are sent at the rate of Transport 2 while the other half are sent at the faster rate of Transport 1. This has the effect of sending the data at twice the rate of Transport 2.

Generally, the transmission time for a system that is Transport Limited is

\[
Tr_i = \left( \frac{1}{N} \right) \left( \frac{P}{S} \right) T_i(i, S)
\]

with a transmission rate of

\[
Rate = N \ast \frac{S}{T_i(i, S)}
\]

where \( i \) corresponds to the slowest transport.

Another possibility is an IML system in which either the sender or receiver has a total process time for all of the transports that is greater than the transport time for any of its transports, as in Figure 4. In this example the transmission time is limited by the processing time of either the sender or the receiver. In cases where the system is Sender Process Limited or Receiver Process Limited the transmission time is

\[
Tr_i = \left( \frac{1}{N} \right) \left( \frac{P}{S} \right) Max \left\{ \sum_{x=0}^{N} S_i(x, S), R_i(x, S) \right\}
\]

with a transmission rate of

\[
Rate = N \ast \frac{S}{Max \left\{ \sum_{x=0}^{N} S_i(x, S), R_i(x, S) \right\}}
\]

B. Optimal Scheduling

The most efficient scheduling should keep the transmitters as busy as possible. This is achieved by using the transports in proportion to their raw throughput rate, or the inverse of their transmission time, causing more segments to be sent across faster transports. With optimal scheduling all transports are used to their fullest capacity during transmission, while Round Robin scheduling only uses each transport at the capacity of the slowest transport.

A system that uses optimal scheduling and is Transport Limited has a transmission time of

\[
Tr_i = Pr(i) \left( \frac{P}{S} \right) T_i(i, S)
\]

where the optimal proportionality is

\[
Pr(i) = \left( \frac{1}{T_i(i, S)} \right) \left( \frac{1}{\sum_{x=0}^{N} \frac{1}{T_i(x, S)}} \right)
\]

and has a rate

\[
Rate = \left( \frac{S}{T_i(i, S)} \right) \left( \frac{1}{Pr(i)} \right)
\]

where \( i \) corresponds to the transport that will yield the highest transmission time (although this equation will give almost the same results for any transport in the IML system).

Proportionality\((i)\) refers to the fraction of total segments that are to be sent over transport \( i \). In a system in which the transport times may dynamically change, the proportionality should also be changed accordingly. WRR and WR scheduling each deal with this proportionality differently.

WRR scheduling sends a ratio of segments on each transport. This weighted ratio should be set as close as practical to the optimal proportionality of that transport. The equations for transmission time and rate can be derived by setting Proportionality\((i)\) to the weighted ratio of transport \( i \).

WR scheduling uses a weighted probability to determine which transport to use for each transmission. This allows a proportionality that is not limited to ratios of integers as WRR is. Each transport would use a weighted probability equal to the optimal proportionality. The equations for transmission time and rate can be derived by setting Proportionality\((i)\) to the weighted probability.

In IML systems that are Process Limited, the transmission time is

\[
Pr(i) = \left( \frac{1}{T_i(i, S)} \right) \left( \frac{1}{\sum_{x=0}^{N} \frac{1}{T_i(x, S)}} \right)
\]

and has a rate

\[
Rate = \left( \frac{S}{T_i(i, S)} \right) \left( \frac{1}{Pr(i)} \right)
\]

where \( i \) corresponds to the transport that will yield the highest transmission time (although this equation will give almost the same results for any transport in the IML system).

**References**

1. For simplicity, the models use the same segment size for all transports. In actual systems, segments may be split into smaller packets or multiple segments may be lumped together into a larger packet for transmission.
Adapters, through the Windows IrDA socket abstraction, which achieved a throughput of 740 Kbps.

Note: While the IrDA adapters use Fast Infrared (FIR) encoding, allowing a transmission rate of up to 4 Mbps, the throughput in this experiment was limited by excessive link turnaround latency caused by a small window size. The Windows IrDA stack defaults to a window size of one, forcing the receiver to acknowledge receipt of a packet before the next packet can be sent. In the case of this experiment, the total time needed to send one packet was 21 ms (4 ms transmission time + 17 ms turnaround latency). Since each packet carries at most 2045 bytes, this yields a maximum throughput of (2045 * 8 bits)/(21 ms) = 780 Kbps. The measured bit rate was 740 Kbps; the 5% difference may be attributed to packet overhead and channel maintenance.

WRR Scheduling was used to assign different proportions to IrDA. The optimal proportionality for IrDA from Equation 6 is

\[
\frac{\text{RawIrDA throughput}}{\text{RawIrDA throughput} + \text{RawBluetooth throughput}} = 740 \left/ \left(740 + 9685\right) \right. = 56.9\%.
\]

The measured throughput values were compared to the values predicted by models for both a Process Limited system and a Transport Limited system, as is shown in Figures 5 and 6. The Transport Limited model predicted the measured throughput of the system with a very high accuracy (within 2%). Optimal Scheduling was approached by using a ratio of BT:IR = 7:9, which corresponds to an IR proportionality of \((9 / (9 + 7)) = 56.3\%\). This gave us a throughput of 1.28 Mbps, or in other words, increased the throughput of the system 73% over what would be achieved using IrDA alone.

Our implementation was clearly Transport Limited rather than Process Limited; an analysis assuming a Process Limited system had little correlation to the actual measured data. This is due to the fact that our sender and receiver were both PC’s. When dealing with embedded devices with limited processing abilities and specialized wireless I/O, the system is more likely to be Process Limited, and it is less likely that adding transports and using inverse multiplexing will be beneficial. In the future the processing speeds for embedded devices will increase.

### V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

Our implementation used two short-range wireless transports: Bluetooth and IrDA. The Bluetooth transport was implemented with Motorola Bluetooth Adapter DIG 442-5 cards, which achieved a throughput of 560 Kbps. The IrDA transport was implemented with ESI XTNDAccess IrDA USB 9685

Adapters, through the Windows IrDA socket abstraction, which achieved a throughput of 740 Kbps.

Note: While the IrDA adapters use Fast Infrared (FIR) encoding, allowing a transmission rate of up to 4 Mbps, the throughput in this experiment was limited by excessive link turnaround latency caused by a small window size. The Windows IrDA stack defaults to a window size of one, forcing the receiver to acknowledge receipt of a packet before the next packet can be sent. In the case of this experiment, the total time needed to send one packet was 21 ms (4 ms transmission time + 17 ms turnaround latency). Since each packet carries at most 2045 bytes, this yields a maximum throughput of (2045 * 8 bits)/(21 ms) = 780 Kbps. The measured bit rate was 740 Kbps; the 5% difference may be attributed to packet overhead and channel maintenance.

WRR Scheduling was used to assign different proportions to IrDA. The optimal proportionality for IrDA from Equation 6 is

\[
\frac{\text{RawIrDA throughput}}{\text{RawIrDA throughput} + \text{RawBluetooth throughput}} = 740 \left/ \left(740 + 9685\right) \right. = 56.9\%.
\]

The measured throughput values were compared to the values predicted by models for both a Process Limited system and a Transport Limited system, as is shown in Figures 5 and 6. The Transport Limited model predicted the measured throughput of the system with a very high accuracy (within 2%). Optimal Scheduling was approached by using a ratio of BT:IR = 7:9, which corresponds to an IR proportionality of \((9 / (9 + 7)) = 56.3\%\). This gave us a throughput of 1.28 Mbps, or in other words, increased the throughput of the system 73% over what would be achieved using IrDA alone.

Our implementation was clearly Transport Limited rather than Process Limited; an analysis assuming a Process Limited system had little correlation to the actual measured data. This is due to the fact that our sender and receiver were both PC’s. When dealing with embedded devices with limited processing abilities and specialized wireless I/O, the system is more likely to be Process Limited, and it is less likely that adding transports and using inverse multiplexing will be beneficial. In the future the processing speeds for embedded devices will increase.

---

### C. Conclusions from Mathematical Analysis

While IM will significantly improve available bandwidth for many devices, it will not do so for all devices. Adding transports may significantly slow the transmission rate for an IML system using Round Robin scheduling (Equation 2). For systems using optimal scheduling, a slower transport may be severely underutilized due to low Proportionality (Equation 6). One should use the formulas above to perform cost evaluation of adding new transports.

Transports that have a transmission time that is shorter than the required processing time will always be Process Limited and should not use Inverse Multiplexing. Adding new transports to an IML system will eventually cause the system to become a Process Limited system and may increase the time needed to process segments as well as causing the transmission rate to decrease according to Equations (4) and (9).

### V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

Our implementation used two short-range wireless transports: Bluetooth and IrDA. The Bluetooth transport was implemented with Motorola Bluetooth Adapter DIG 442-5 cards, which achieved a throughput of 560 Kbps. The IrDA transport was implemented with ESI XTNDAccess IrDA USB 9685

Adapters, through the Windows IrDA socket abstraction, which achieved a throughput of 740 Kbps.

Note: While the IrDA adapters use Fast Infrared (FIR) encoding, allowing a transmission rate of up to 4 Mbps, the throughput in this experiment was limited by excessive link turnaround latency caused by a small window size. The Windows IrDA stack defaults to a window size of one, forcing the receiver to acknowledge receipt of a packet before the next packet can be sent. In the case of this experiment, the total time needed to send one packet was 21 ms (4 ms transmission time + 17 ms turnaround latency). Since each packet carries at most 2045 bytes, this yields a maximum throughput of (2045 * 8 bits)/(21 ms) = 780 Kbps. The measured bit rate was 740 Kbps; the 5% difference may be attributed to packet overhead and channel maintenance.

WRR Scheduling was used to assign different proportions to IrDA. The optimal proportionality for IrDA from Equation 6 is

\[
\frac{\text{RawIrDA throughput}}{\text{RawIrDA throughput} + \text{RawBluetooth throughput}} = 740 \left/ \left(740 + 9685\right) \right. = 56.9\%.
\]

The measured throughput values were compared to the values predicted by models for both a Process Limited system and a Transport Limited system, as is shown in Figures 5 and 6. The Transport Limited model predicted the measured throughput of the system with a very high accuracy (within 2%). Optimal Scheduling was approached by using a ratio of BT:IR = 7:9, which corresponds to an IR proportionality of \((9 / (9 + 7)) = 56.3\%\). This gave us a throughput of 1.28 Mbps, or in other words, increased the throughput of the system 73% over what would be achieved using IrDA alone.

Our implementation was clearly Transport Limited rather than Process Limited; an analysis assuming a Process Limited system had little correlation to the actual measured data. This is due to the fact that our sender and receiver were both PC’s. When dealing with embedded devices with limited processing abilities and specialized wireless I/O, the system is more likely to be Process Limited, and it is less likely that adding transports and using inverse multiplexing will be beneficial. In the future the processing speeds for embedded devices will increase.
the same, making Inverse Multiplexing an attractive option.

VI. Future Work and Related Topics

The following research areas need to be explored with respect to Inverse Multiplexing:

- **Optimal IM Policies.** The policies governing use of IM should address more than just throughput. Disparate transports vary in their power requirements, security, resistance to interference, latency, packet size and other factors. The policies of transport selection and scheduling should be governed by the requirements of the device and usage model [8].

- **Quality of Transport (QoT).** On-going research in QoT attempts to ensure the best connection possible between devices when multiple transports are available [9]. Inverse Multiplexing can be integrated into the QoT architecture to seamlessly provide high bandwidth when appropriate.

- **Quality of Service (QoS).** Not all data traffic is equally impacted by network latency. QoS appears to be even more critical in wireless networks than in wired environments [10], because of the lower bandwidth, higher error rates caused by interference, and higher delay and jitter. Using heterogeneous transports we can take advantage of the characteristics of each. For example, we can give priority to real-time applications by exploiting the higher bandwidth that IrDA offers over Bluetooth, when it is available.

- **Scalability.** Adding multiple transports to any device will ultimately cause that system to become Process Limited, negating the value of adding additional transports. Another limit on IM scalability is the mutual interference caused by multiple transceivers using the same medium in the same area. Multiple IrDA connections cannot effectively share the same space without significant modification to the transceivers, such as wavelength or polarization filtering. Overlapping Bluetooth piconets may experience significant mutual interference, which can be partially alleviated with synchronization [11][12]. Transceivers that mutually interfere will eventually affect each other’s performance to the point that their addition to a system will actually degrade overall system performance.

- **Improved Device Discovery.** Multiple transports may be used to improve latency inherent with device discovery. For example, IrDA device discovery is much quicker than Bluetooth inquiry and can be used to quickly establish a connection [13] when a device is equipped with both technologies. Bluetooth device discovery using multiple Bluetooth transceivers may be quicker, since multiple simultaneous inquiry processes would more effectively search the frequency band used by Bluetooth.

- **Multi-transport Usage Models.** Certain usage models are best satisfied using multiple short-range wireless transports. A device with multiple transports may be used as a bridge between devices that do not have transports in common. Multiple transports may also allow asymmetric links to conserve power usage. For example, a power-constrained device may receive data on a transport that requires significantly more energy to send data than to receive, such as diffuse infrared. The device could then send acknowledgements across a less demanding transport, such as Bluetooth.

VII. Conclusion

In this paper we have shown that Inverse Multiplexing is a practical way to increase the bandwidth of short-range wireless systems that are Transport Limited. The techniques and scheduling algorithms used in wired Inverse Multiplexing can be adapted for use with short-range wireless transports. The mathematical analysis examined the tradeoffs of several scheduling algorithms and showed that optimal scheduling could be approached using Weighted Round Robin scheduling. Models were made for both Transport Limited and Process Limited systems, and it was shown that Inverse Multiplexing was impractical for Process Limited systems such as resource-constrained embedded devices. The practicality of Inverse Multiplexing for Transport Limited systems (such as Laptops, PCs, and high-end embedded devices) was clearly shown both by mathematical analysis and by empirical experimentation using Bluetooth and IrDA.
REFERENCES


