The ATM Forum

Technical Committee

ATM Forum Performance Testing Specification

AF-TEST-TM-0131.000

October, 1999

© 1999 by The ATM Forum. The ATM Forum hereby grants its members the limited right to reproduce in whole, but not in part, this specification for its members internal use only and not for further distribution. This right shall not be, and is not, transferable. All other rights reserved. Except as expressly stated in this notice, no part of this document may be reproduced or transmitted in any form or by any means, or stored in any information storage and retrieval system, without the prior written permission of The ATM Forum.

The information in this publication is believed to be accurate as of its publication date. Such information is subject to change without notice and The ATM Forum is not responsible for any errors. The ATM Forum does not assume any responsibility to update or correct any information in this publication. Notwithstanding anything to the contrary, neither The ATM Forum nor the publisher make any representation or warranty, expressed or implied, concerning the completeness, accuracy, or applicability of any information contained in this publication. No liability of any kind shall be assumed by The ATM Forum or the publisher as a result of reliance upon any information contained in this publication.

The receipt or any use of this document or its contents does not in any way create by implication or otherwise:

- Any express or implied license or right to or under any ATM Forum member company's patent, copyright, trademark or trade secret rights which are or may be associated with the ideas, techniques, concepts or expressions contained herein; nor

- Any warranty or representation that any ATM Forum member companies will announce any product(s) and/or service(s) related thereto, or if such announcements are made, that such announced product(s) and/or service(s) embody any or all of the ideas, technologies, or concepts contained herein; nor

- Any form of relationship between any ATM Forum member companies and the recipient or user of this document.

Implementation or use of specific ATM standards or recommendations and ATM Forum specifications will be voluntary, and no company shall agree or be obliged to implement them by virtue of participation in The ATM Forum.

The ATM Forum is a non-profit international organization accelerating industry cooperation on ATM technology. The ATM Forum does not, expressly or otherwise, endorse or promote any specific products or services.

NOTE: The user's attention is called to the possibility that implementation of the ATM interoperability specification contained herein may require use of an invention covered by patent rights held by ATM Forum Member companies or others. By publication of this ATM interoperability specification, no position is taken by The ATM Forum with respect to validity of any patent claims or of any patent rights related thereto or the ability to obtain the license to use such rights. ATM Forum Member companies agree to grant licenses under the relevant patents they own on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. For additional information contact:

The ATM Forum
Worldwide Headquarters
2570 West El Camino Real, Suite 304
Mountain View, CA 94040-1313
Tel: +1-650-949-6700
Fax:+1-650-949-6705

Acknowledgement

Much work went into the development of this specification. It could not have been completed without the ATM Forum contributions and the participation of many people in the TEST and TM working groups. In particular, the Editor would like to recognize the following members who made significant contributions.

John Adams
Mustapha Aissaoui
Gojko Babic
Abdella Battou
J. Belanger
Arnold Bragg
Walter Buehler
Norman Carder
Tom Chen
John Clark
Leslie Collica
Gregan Crawford (TEST-chair)
Siamak Dastangoo
Emmanuel Desmet
Arjan Durresi
Rod Ferguson
G. Garner
Natalie Giroux (TM-chair)
Kenneth Green
Kenneth C. Glossbrenner
Kent Headrick
Ivy Hsu
Werner Hug
Doug Hunt
Raj Jain (former Editor)
Sungwon Kang
Deepak Kataria
Dinyar Kavouspour
Hyoung-soo Kim
Kuo-Hui Liu
Brian McBride
Ken McInerney
Randy Mitchell
B. Morris
Raj Nair
Bruce Northcote
Deane Osbourne
Rick Raskin
Andreas Schubert
Nancy Schult
Piergiorgio Vittori
Tom Worster

The assistance by these members and the many others that contributed to getting this work started and participated in the TEST and TM working groups is greatly appreciated.

Fred Kaudel, Editor.

Table of Contents

1. Introduction

Performance testing in ATM deals with the measurement of the level of quality of a System Under Test (SUT) or an Implementation Under Test (IUT) under well-known conditions. The level of quality can be expressed in the form of metrics such as latency, end-to-end delay, and effective throughput. Performance testing can be carried at the end-user application level (*e.g.*, FTP, NFS), at or above the ATM layers (*e.g.*, cell switching, signalling). Performance testing also describes in details the procedures for testing the IUTs in the form of test suites. These procedures are intended to test the SUT or IUT and do not assume or imply any specific implementation or architecture of these systems.

This document highlights the objectives of performance testing and suggests an approach for the development of the test suites. Several test cases are embedded into measurement procedures of the different metrics. This specification is not intended to be a complete test suite.

1.1. Scope

Asynchronous Transfer Mode, as an enabling technology for the integration of services, is gaining an increasing interest and popularity. ATM networks are being progressively deployed and in most cases a smooth migration to ATM is prescribed. This means that most of the existing applications can still operate over ATM via service emulation or service interworking along with the proper adaptation of data formats. At the same time, several new applications are being developed to take full advantage of the capabilities of the ATM technology through an Application Programming Interface (API).

While ATM provides an elegant solution to the integration of services and allows for high levels of scalability, the performance of a given application may vary substantially with the IUT or the SUT utilized. The variation in the performance is due to the complexity of the dynamic interaction between the different layers. For example, an application running with TCP/IP stacks will yield different levels of performance depending on the interaction of the TCP window flow control mechanism and the ATM network congestion control mechanism used. Hence, the following points and recommendations are made. First, ATM adopters need guidelines on the measurement of the performance of user applications over different systems. Second, some functions above the ATM layer, *e.g.*, adaptation, signalling, constitute applications (*i.e.*, IUTs) and as such should be considered for performance testing. Also, it is essential that these layers be implemented in compliance with the ATM Forum specifications. Third, performance testing can be executed at the ATM layer in relation to the QoS provided by the different service categories. Finally, because of the extensive list of available applications, it is preferable to group applications in generic classes. Each class of applications requires different testing environment such as metrics, test suites and traffic test patterns. Note that the same application, *e.g.*, ftp, can yield different performance results depending on the underlying layers used (TCP/IP to ATM versus TCP/IP to MAC layer to ATM). Thus, performance results should be compared based on the utilization of the same protocol stack.

Performance testing is related to user perceived performance of ATM technology. In other words, goodness of ATM will be measured not only by cell-level performance but also by frame-level performance and performance perceived at higher layers.

Most of the quality of Service (QoS) metrics, such as cell transfer delay (CTD), cell delay variation (CDV), cell loss ratio (CLR), and so on, may or may not be reflected directly in the performance perceived by the user. For example, while comparing two switches if one gives a CLR of 0.1% and a frame loss ratio of 0.1% while the other gives a CLR 1% but a frame loss of 0.05%, the second switch will be considered superior by many users.

The ATM Forum and ITU-T have standardized the definitions of ATM layer QoS metrics and their measurement [6, 2, 1, 9]. This specification does the same for adaptation layer performance metrics. Without a standard definition, each vendor will use their own definition of common metrics such as throughput and latency resulting in confusion in the market place. Avoiding such confusion will help buyers, eventually lead to better sales and result in the success of ATM technology.

1.2. Goals of Performance Testing

The goal of this effort is to enhance the marketability of ATM technology and equipment. Any additional criteria that help in achieving that goal can be added later to this list.

a. The ATM Forum shall define metrics that will help compare various ATM equipment in terms of performance.
b. The metrics shall be such that they are independent of switch or NIC architecture. The same metrics shall apply to all architectures.
c. The metrics can be used to help predict the performance of an application or to design a network configuration to meet specific performance objectives.
d. The ATM Forum will develop a precise methodology for measuring these metrics. The methodology will include a set of configurations and traffic patterns that will allow vendors as well as users to conduct their own measurements.
e. The testing shall cover all classes of service including CBR, rt-VBR, nrt-VBR, UBR, ABR, and GFR.
f. The metrics and methodology for different service categories may be different.
g. The testing should cover as many protocol stacks and ATM services as possible. As an example, measurements for verifying the performance of services such as IP, Frame Relay and SMDS over ATM may be included.
h. The following objectives are set for ATM performance testing:
   (i) Definition of criteria to be used to distinguish classes of applications.
   (ii) Definition of classes of applications, at or above the ATM Layer, for which performance metrics are to be provided.
   (iii) Identification of the functions at or above the ATM Layer which influence the perceived performance of a given class of applications. Examples of such functions include traffic shaping, quality of service, and adaptation. These functions need to be measured in order to assess the performance of the applications within that class.
   (iv) Definition of common performance metrics for the assessment of the performance of all applications within a class. The metrics should reflect the effect of the functions identified in (iii).

i. The scope of this first revision of the ATM Forum Performance Testing Specification is limited to AAL-5.

1.3. Non-Goals of Performance Testing

a. The ATM Forum is not responsible for conducting any measurements.
b. The ATM Forum will not:
   (i) certify measurements.
   (ii) evaluate or assess results obtained by companies or other bodies.
   (iii) certify bodies conducting measurements.
c. The ATM Forum will not set thresholds such that equipment performing below those thresholds are called "unsatisfactory."
d. The ATM Forum neither performs nor specifies benchmark testing (see [1]).
e. The ATM Forum will not establish any requirement that dictates a cost versus performance ratio.
f. Applications whose performance cannot be assessed by common implementation independent metrics are excluded from the scope of ATM performance testing. In this case, performance is tightly related to the implementation. An example of such applications is network management, whose performance depends on whether it is a centralized or a distributed implementation.

1.4. Terminology

The following definitions are used in this document:

- *Activity*: A phase consists of one or more active periods, separated by idle periods (inter-activity gaps).
- *ATM Analyzer*: ATM measuring equipment used to measure the characteristics of traffic received from the SUT.
- *ATM Connection*: An ATM connection consists of the concatenation of ATM layer links in order to provide an end-to-end transfer capability to the access point (or T reference point) - from ITU-T Recommendation I.150 [3].
- *ATM Generator*: ATM measuring equipment used to produce traffic with specified characteristics.
- *ATM Interface*: The ATM physical interface where ATM traffic enters and/or leaves the SUT.
- *Background Connection*: An ATM connection that carries background traffic.
- *Background Traffic*: Traffic made up of cells whose purpose is to load the SUT at an appropriate level but the performance of these cells is not of primary interest.
- *Connection Load*: Precisely defined specification of a pattern of ATM cells over a single connection. When cells are parts of frames, a connection load may be defined in term frame pattern and inter-frame gaps distribution.
- *Foreground Connection*: An ATM connection that carries foreground traffic.
- *Foreground Traffic*: Traffic made up of cells or frames whose performance is being measured.
- *Frame*: A sequence of cells corresponding to an AAL-5 PDU (delineated by setting the AUU bit to 1 in the last cell of the frame). The AUU bit is the LSB in the three-bit PTI code point 0xx (user data cell, as defined in ITU-T Recommendation I.361 [8]).

- *Frame Pattern*: Precisely defined pattern of cells and inter-cell gaps, inter-frame gaps, and frame sizes. A frame pattern can be defined statistically in terms of distributions of frame sizes, inter-frame gaps, and inter-cell gaps.
- *Frame Size*: The number of cells in a frame.
- *Implementation Under Test (IUT)*: The part of the system that is to be tested.
- *Input ATM Interface Rate*: Maximum nominal rate in cells/second at which the cells can be received at the interface.
- *Interface Load*: Set of Connection Loads applied to an input ATM interface.
- *Interval*: An active period consists of one or more characteristic intervals, separated by inter-interval gaps.
- *Inter-activity Gap*: The number of idle cells between the last interval of one activity and the first interval of the next activity.
- *Inter-cell Gap*: The time between transmission of the last bit of one cell and transmission of the first bit of the next cell of the same frame.
- *Inter-frame Gap*: The time between transmission of the last bit of the last cell one frame and transmission of the first bit of the first cell of the next frame.
- *Inter-interval Gap*: The number of cells between transmission of the last frame of one interval and the first frame of the next interval.
- *Inter-phase Gap*: The number of cells between the last activity of one phase and the first activity of the next phase.
- *Loopback*: An external connection that connects the output and input of the same ATM interface.
- *Measurement Point*: A measurement point is located at an interface that separates either customer equipment/customer network or a Switching/Signalling Node from an attached transmission system at which protocols can be observed [5].
- *Metric*: A quantitative measure of the goodness of overall service offered by an SUT, for example, throughput, throughput fairness. It reflects quantitatively the response or the behavior of an IUT or an SUT.
- *Monitor*: ATM measuring equipment used to assess transfer performance of an SUT [9].
- *Network Module*: A group of switch ATM interfaces that physically reside on a single card.
- *Output ATM Interface Rate*: The maximum nominal rate in cells/second at which the cells can be transmitted from the interface.
- *Parameter*: A quantitative measure of the goodness of services received by a connection, for example, cell transfer delay, cell delay variation.
- *Phase*: A test consists of one or more test phases, separated by inter-phase gaps.
- *Port*: Same as ATM Interface.
- *Reference Load*: The set of Interface Loads applied to an SUT.
- *Scalable Configuration*: A configuration that permits loading the SUT using a minimal number of ATM monitors.
- *Switch Fabric*: The switch component whose main function is to transfer cells among various interfaces of the switch.
- *System Under Test (SUT)*: Any collection of ATM equipment that is being tested. It could be a single switch or a network of ATM switches. It includes the IUT.
- *Test Case*: A series of test steps needed to put an IUT into a given state to observe and describe its behavior.

- *Test Suite*: A complete set of test cases, possibly combined into nested test groups, that is necessary to perform testing for an IUT or a protocol within an IUT.
- *Traffic Load*: Connection Load, Interface Load or Reference Load.
- *Wire*: The physical medium used for external connections of the SUT's ATM interfaces. The medium could be copper cables, optical fibers, or wireless links.

1.5. Abbreviations

AAL	ATM Adaptation Layer
ABR	Available Bit Rate
AG	Application Goodput
API	Application Programming Interface
ATM	Asynchronous Transfer Mode
AUU	ATM User to User
BEDC	Block Error Detection Code
BIP	Bit Interleaved Parity
BLER	BLock Error Result
BR	Burst Responsiveness
CAC	Connection Admission Control
CBR	Constant Bit Rate
CDV	Cell Delay Variation
CDVT	Cell Delay Variation Tolerance
CER	Cell Error Ratio
CIT	Cell Input Time
CLP	Cell Loss Priority
CLR	Cell Loss Ratio
CMR	Cell Misinsertion Rate
COT	Cell Output Time
CRC	Cyclic Redundancy Check
CRE	Cell-level Reference Event
CS	Convergence Sublayer
CSR	Cell misSequenced Ratio
CTD	Cell Transfer Delay
EDC	Error Detection Code
EOF	End of Frame
FFL	Full Foreground Load
FIFO	First In, First Out
FILO	First In, Last Out
FLR	Frame Loss Ratio
FRE	Frame-level Reference Event
FSN	Frame Sequence Number
FTP	File Transfer Protocol
GFR	Guaranteed Frame Rate
HEC	Header Error Control
HW	Hardware
ICG	Inter-Cell Gap
IETF	Internet Engineering Task Force

IFG	Inter-Frame Gap
iid	independent, identically distributed
IP	Internet Protocol
IPR	Impaired PDU Ratio
IRE	Internal Reference Event
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union - Telecommunication standardization sector
IUT	Implementation Under Test
LAN	Local Area Network
LIFO	Last In, First Out
LILO	Last In, Last Out
LSB	Least Significant Bit
MAC	Media Access Control
MaxCTD	Maximum Cell Transfer Delay
MBL	Maximum Background Load
MBS	Maximum Burst Size
MCBS	Maximum Cell Burst Size
MCR	Minimum Cell Rate
MCSN	Monitoring Cell Sequence Number
MCTD	Mean Cell Transfer Delay
MFBS	Maximum Frame Burst Size
MFL	Maximum Foreground Load
MFS	Maximum Frame Size
MIMO	Message In, Message Out
MinCTD	Minimum Cell Transfer Delay
MP	Measurement Point
MSB	Most Significant Bit
MTU	Maximum Transmission Unit
NFS	Network File System
NIC	Network Interface Card
NP	Network Performance
NPC	Network Parameter Control
OAM	Operations and Maintenance
PCR	Peak Cell Rate
PDU	Protocol Data Unit
PPI	Proprietary Payload Indicator
PRBS	Pseudo Random Bit Sequence
PTI	Payload Type Indicator
PVC	Permanent Virtual Circuit
QoS	Quality of Service
RL	Reference Load
RLM	Reference Load Model
SAP	Service Access Point
SAR	Segmentation And Reassembly

SCR	Sustained Cell Rate
SECBR	Severely-Errored Cell Block Ratio
SMDS	Switched Multi-Megabit Data Service
SUT	System Under Test
SVC	Switched Virtual Circuit
SW	Software
TCP	Transport Control Protocol
TCPT	Test Cell Payload Type
TMN	Telecommunications Management Network
TRCC	Total Received Cell Count
TUC	Total User Cell
UBR	Unspecified Bit Rate
UN	UNspecified bytes
UNI	User Network Interface
UPC	Usage Parameter Control
UTP	Unshielded Twisted Pair
nrt-VBR	non-real-time Variable Bit Rate
rt-VBR	real-time Variable Bit Rate
VC	Virtual Circuit
VCI	Virtual Channel Identifier
VCC	Virtual Channel Connection
VP	Virtual Path
VPC	Virtual Path Connection
VPI	Virtual Path Identifier
WAN	Wide Area Network
WG	Working Group

2. Overview of Performance Testing

Applications should be grouped to simplify their testing. Applications with similar performance requirements can be grouped together. This simplifies the testing of new applications, as they most likely fit into an existing grouping, for which the performance metrics and test procedures of this specification can be used. This section provides overviews of performance testing above the ATM layer, performance testing at the ATM layer, and requirements for performance testing.

2.1. Performance Testing Above the ATM Layer

Performance metrics can be measured at the user application layer, and sometimes at the transport layer and the network layer, and can give an accurate assessment of the perceived performance.

The perceived performance of a user application running over an ATM network is dependent on many parameters. It can vary substantially by changing an underlying protocol stack, the ATM service category it uses, the congestion control mechanism used in the ATM network, *etc.* Furthermore, there is no direct and unique relationship between the ATM Layer Quality of Service (QoS) parameters and the perceived application performance. For example, in an ATM network implementing a packet-level discard congestion mechanism, applications using TCP as the transport protocol may see their effective throughput improved while the measured cell loss ratio may be relatively high. In practice, it is difficult to carry out measurements in all the layers that span the region between the ATM Layer and the user application layer given the inaccessibility of testing points. More effort needs to be invested to define the performance at these layers. These layers include adaptation, and signalling. This specification applies to AAL-5.

2.2. Performance Testing at the ATM Layer

The notion of application at the ATM Layer is related to the service categories provided by the ATM service architecture. The Traffic Management Specification, Version 4.1 [2] specifies six service categories: CBR, rt-VBR, nrt-VBR, UBR, ABR, and GFR. Each service category defines a relation of the traffic characteristics and the Quality of Service (QoS) requirements to network behavior. There is an assessment criteria of the QoS associated with each of these parameters. These are summarized below.

QoS PERFORMANCE PARAMETER	*QoS ASSESSMENT CRITERIA*
Cell Error Ratio	Accuracy
Severely-Errored Cell Block Ratio	Accuracy
Cell Misinsertion Ratio	Accuracy
Cell Loss Rate	Dependability
Cell Transfer Delay	Latency
Cell Delay Variation	Accuracy

Section 5.6 of ITU-T Recommendation I.356 [6] further defines the Severely-Errored Cell Block Ratio.

Performance testing at the ATM Layer deals with in-service and out-of-service measurements of the QoS parameters for all six service categories (or application classes in the context of performance testing): CBR, rt-VBR, nrt-VBR, UBR, ABR and GFR.

The following cases are covered:

(i) Performance of the SUT under non-overloaded conditions.

(ii) Performance of the SUT under overload conditions. In this case, the efficiency of the congestion avoidance and congestion control mechanisms of the SUT are tested.

Appendix D defines methods and configurations for both in-service and out-of-service measurements. The in-service mode uses OAM cells, while the out-of-service mode defines the payloads to be used for test cells on connections running out-of-service measurements. Measurement methods for both in-service and out-of-service modes are also defined in the ITU-T Recommendation O.191 [9] whereas in-service performance monitoring procedures are specified in ITU-T Recommendation I.610 [12] on OAM principles and functions. ATM Forum specification [1] also defines out-of-service measurement of several QoS parameters. However, detailed test cases and procedures, as well as test configurations are needed for both in-service and out-of-service measurement of QoS parameters.

2.3. Requirements for Performance Testing

To provide common performance metrics that are applicable to a wide range of SUT's and that can be uniquely interpreted, the following requirements must be satisfied:

(i) Reference load models for the six service categories CBR, rt-VBR, nrt-VBR, UBR, ABR, and GFR, and for AAL-5 frame layer are required. Candidate Reference Load Models (RLMs) shall meet the following criteria:

- The RLM should embody the relevant characteristics of the actual traffic. This means that the System Under Test (SUT) should react similarly to RLM or actual traffic with respect to the important metrics (*e.g.,* buffer characteristics, delays, cell loss rate).
- The algorithm for generating a RLM should be precisely defined and feasible to implement.
- The RLM should be scaleable (*i.e.,* the traffic volume can be increased or decreased in a straightforward manner).

The traffic generated by the RLM should be reproducible. Reference load models (also referred to as test traffic profiles) for cell transfer performance measurement of CBR and VBR service categories are defined in ITU-T Recommendation O.191 [9]. Reference load models for other applications should be considered.

(ii) Test cases and configurations must not assume or imply any specific implementation or architecture of the SUT.

3. Performance Testing Methodology

In the following description, Implementation Under Test (IUT) refers to an ATM switch. However, the definitions and measurement procedures are general and may be used for other devices or a network consisting of multiple switches as well. Section 3.1 describes some of the issues when performance testing ATM devices. Section 3.2 defines the Measurement Points between which the Reference Loads, described in Section 3.3, are applied. Section 3.4 describes why a System Under Test (SUT) should always refer to the IUT plus any specific connection configuration that is used to generate the Reference Load. Section 3.5 discusses general measurement procedures. Section 3.6 considers statistical variations. Section 3.7 highlights optional traffic management functions and procedures. Section 3.8 describes result reporting.

3.1. ATM Specific Issues

ATM is a sophisticated communication technology providing Quality of Service (QoS) for users. This is different than the traditional best effort services, and more sophisticated performance testing is required. Some of the issues when performance testing ATM devices are:

1. Connections have a specified set of traffic and quality of service parameters. These parameters are defined so that a connection can obtain some guaranteed performance or QOS. If the connection is not conformant to its traffic parameters, it is deemed non-conformant and can be subject to Usage Parameter Control. The UPC function may tag or discard cells, thereby affecting performance. Therefore it is imperative that test connections are defined so that they can handle the traffic sources characteristics. Otherwise the

performance reported will not be accurate, since it could be simply deemed as non-conforming.

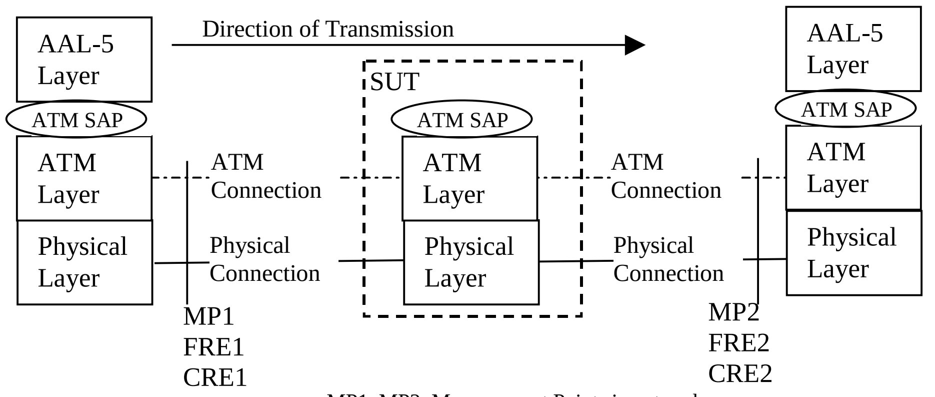
2. The allowable connections on a SUT are determined by a connection admission control (CAC) system. There are generally no performance guarantees on connections that would be rejected by the SUT CAC. Therefore, it is important that only connections allowed by the CAC are tested.
3. ATM supports multiple virtual connections (VC). The performance of an SUT with a single VC generally does not extrapolate to the performance of the SUT with multiple VCs.
4. Many SUTs have proprietary congestion control schemes that can be used to improve performance under certain conditions. The tester should use these performance enhancing schemes if applicable when testing the SUT, and should list them in the test report.
5. Cells can be marked as CLP 1 or CLP 0. The CLP settings on the traffic source can have a significant impact on the performance of the SUT.
6. ATM SUTs generally have significant gains from statistical multiplexing. However it is hard to test the performance gains from statistical multiplexing without a large number of test generators and sophisticated test sources. The scaleable test configurations provided in this specification may not show any performance gains from statistical multiplexing.

3.2. Measurement Points and Reference Events

Following the definition of cell reference events [2], a similar definition for frame reference events is provided. Moreover, it is illustrated that the physical points where the events are measured are the same for both cell events and frame events.

A frame is defined as a sequence of cells corresponding to an AAL-5 PDU (delineated by setting the AUU bit to 1 in the last cell of the frame). The AUU bit is the LSB in the three-bit PTI code point 0xx (user data cell, as defined in ITU-T Recommendation I.361 [8]).

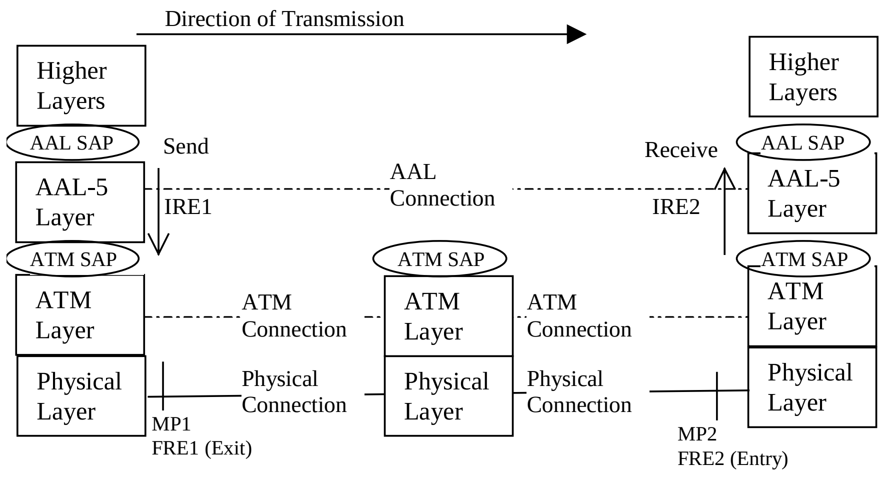
To describe frame-level performance parameters consistent with the general structure in ITU-T Recommendation I.350 [4], it is necessary to define frame-level reference events that are observable at measurement points in a network and then define relevant performance parameters based on these reference events. The measurement points are defined at physical locations in a network. The frame-level reference events are observable at these locations with suitable test sets; that is, the reference event definitions are based on physical layer test access. Figure 3.1 shows two such frame-level reference events that are labeled FRE1 and FRE2, and that are observable with suitable test equipment at the measurement points labeled MP1 and MP2, respectively. The SUT is tested either out-of-service in its network, or in a laboratory.



MP1, MP2: Measurement Points in network
FRE1, FRE2: Frame-level Reference Events measurable at MP1, MP2
CRE1, CRE2: Cell-level Reference Events measurable at MP1, MP2
SAP: Service Access Point

Figure 3.1: Measurement Points and Reference Events

CRE1 and CRE2 are cell-level reference events observable at MP1 and MP2 in Figure 3.1. Cell entry and cell exit events are cell reference events [2]. The AAL-5 layer is the natural place to consider defining reference events at the frame-level. Figure 3.2 illustrates two such frame-level reference events defined inside the AAL-5 layer of each full protocol stack. The protocol stacks illustrated are in the test sets at MP1 and MP2.



MP1, MP2: Measurement Points in network
FRE1, FRE2: Frame-level Reference Events measurable at MP1, MP2
IRE1, IRE2: (Internal) Reference Event internal to indicated protocol stacks
SAP: Service Access Point

Figure 3.2: Frame-Level Internal and External Reference Events and Measurement Points

Reference events occur as relevant PDUs move across the protocol layer interface between the AAL-5 Convergence Sublayer (CS) and the AAL-5 Segmentation And Reassembly (SAR) sublayer. With respect to the indicated direction of transmission, these are labeled AAL-5 Internal Reference Event 1 (IRE1) and AAL-5 Internal Reference Event 2 (IRE2), respectively. Ideally, the IREs are the reference events one would like to measure to determine frame-level performance. However, since the IREs reference interfaces within the protocol stack, they are generally not physically accessible with test sets. Each FRE is defined to approximate an IRE, and is observable at a MP with a suitable test set (*i.e.*, the FRE is defined based on physical layer test access). The information content of FRE1 is, for practical purposes, the same as that of the corresponding IRE1 that generated it. The time of occurrence of FRE1, T1, lags behind the occurrence of IRE1 by a small and quantifiable amount. Similarly, the time of occurrence of FRE2, T2, leads the occurrence of IRE2 by a small and quantifiable amount.

Consistent with the structure provided by ITU-T Recommendation I.350 [4] requiring that performance parameters be defined in terms of performance-significant reference events that are observable at MPs, FRE1 and FRE2 shown in Figure 3.1 fulfill the requirement of observability at network measurement points MP1 and MP2 (located on the network side of these AAL-5 SAPs, and as close as practical to them), but IRE1 and IRE2 shown in Figure 3.2 do not.

Using the definitions of cell entry and exit events in [2], the following definitions for two frame-level reference events are proposed:

- Frame-level Reference exit Event (shown as FRE1): the occurrence of the cell exit event for the first user data cell of the frame;
- Frame-level Reference entry Event (shown as FRE2): the occurrence of the cell entry event for the last user data cell of the frame.

Observation of these frame-level reference events is possible because the MPs are located in a network. As shown in Figure 3.1, MP1 is located near the transmitting equipment and MP2 is located near the receiving equipment.

Test equipment at MP1 and MP2 would reconstruct the frames. Since each of these MPs is located near an ATM SAP, the cell transfer reference events CRE1 and CRE2 as defined in ITU-T Recommendation I.353 [5] are also observable, and hence, these MPs can be used to measure ATM cell transfer performance parameters. This coincidence of MPs for frame-level performance and for ATM cell transfer performance simplifies identifying the relations between these two types of performance parameters.

The frame-level performance parameters can be defined based on the above frame-level reference events. Following the approach of ITU-T Recommendation I.356 [6], appropriate frame transfer outcomes are defined based upon the occurrence of appropriate frame-level reference entry (FRE2) events at an MP2 near receiving equipment that corresponds to frame-level reference exit (FRE1) events at an MP1 near transmitting equipment. Two frame-level reference events correspond if they are created by the same frame.

Again, in parallel with ITU-T Recommendation I.356 [6], a frame transfer outcome is defined as the occurrence at an MP2 of a FRE2 corresponding to the occurrence at an MP1 of a FRE1, within a specified time T_{max}. A frame transfer outcome would generally be further classified by certain criteria, such as whether or not the user information bits in the FRE2 match the user information bits in the corresponding FRE1.

Consistent with ITU-T draft Recommendation X.144 Amendment 1 Annex C [14], this approach will be demonstrated by applying it in sections 4.5.1 and 4.3.1 to develop proposed definitions for user information loss and user information delay performance parameters associated with frame transfer outcomes, respectively.

3.3. Reference Loads

A prerequisite for successful and repeatable performance testing is the definition of standard Reference Load Models (RLMs). RLMs are used to characterize test traffic in a well-defined manner for input into a System Under Test (SUT). With the use of well-defined RLMs specified as part of a test, the tester can run tests that are reproducible by other labs.

Given that this specification discusses measuring AAL-5 performance of ATM systems, it is fitting that a method for defining standardized frame RLMs should be defined for use in that testing.

The following sections define a consistent methodology for defining frame RLMs. This will allow testers to define test sources used for testing, without ambiguity.

3.3.1. Basic Framework of a Frame Reference Load Model

There are two basic logical components for frame RLMs to fully characterize their cell-level traffic patterns. These components are the frame sources and the cell multiplexer. The frame sources and multiplexer are only logical concepts that do not need to exist in any implementation. They can be considered virtual devices.

The frame source creates frames as sequences of cells with a well-defined pattern. Generally each source corresponds to a single VC. The multiplexer in turn describes how multiple frame sources, or perhaps cell sources as well, are put into a single output stream. For simplicity the input cell rate of a frame source to the multiplexer is defined to be equivalent to the output cell rate of the multiplexer.

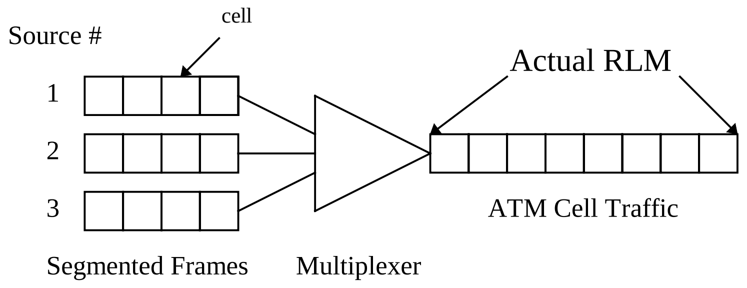


Figure 3.3: Logical model of a frame RLM

The multiplexer conceptually contains cell buffers for each source and some arbitration device. The arbitration device specifies how cells from multiple sources will be placed on the output stream. Conceptually, the actual RLM (well-defined traffic pattern) is what appears at the egress of the multiplexer. The multiple sources and multiplexer are used to characterize the RLM only, and need not exist in any implementation.

Using this framework requires the specification of two sets of parameters. These parameters are the frame source parameters for each source, and the multiplexer parameters.

3.3.2. Frame Source Parameters

The following parameters can be used to characterize frame sources from a variety of applications. These parameters can be specified as having any mathematical, statistical or algorithmic distribution the author of a RLM feels is appropriate. This will allow for a wide range of useful RLMs.

The set of required parameters is the minimal subset to be used for defining a frame source. Theoretically, they can be used to define the majority of input RLMs, however for some sources this may be difficult. The optional parameters are defined recursively so that long term RLMs can be defined more simply.

Table 3.1: Frame Source Parameters

Required Optional	**Parameter**	**Units**	**Comment**
Optional	*Number_Phases*	None	Integer > 0.
Optional	*Inter-Phase Gap*	Cells	Number of idle cells between generation of consecutive phases.
Optional	*Number_Active_Periods*	None	Integer > 0.
Optional	*Inter-Activity Gap*	Cells	Number of idle cells between generation of consecutive activity periods.
Optional	*Number_Intervals*	None	Integer > 0.
Optional	*Inter-Interval Gap*	Cells	Number of idle cells between generation of consecutive intervals.
Optional	*Number_Frames*	None	Integer > 0.
Required	*Inter-Frame Gap*	Cells	Number of idle cells between generation of consecutive frames
Required	*Frame Size*	Cells	Integer > 0.
Required	*Inter-Cell Gap*	Cells	Number of idle cells between generation of consecutive user cells in a frame.

The following diagram illustrates the hierarchical relationship of the frame source parameters.

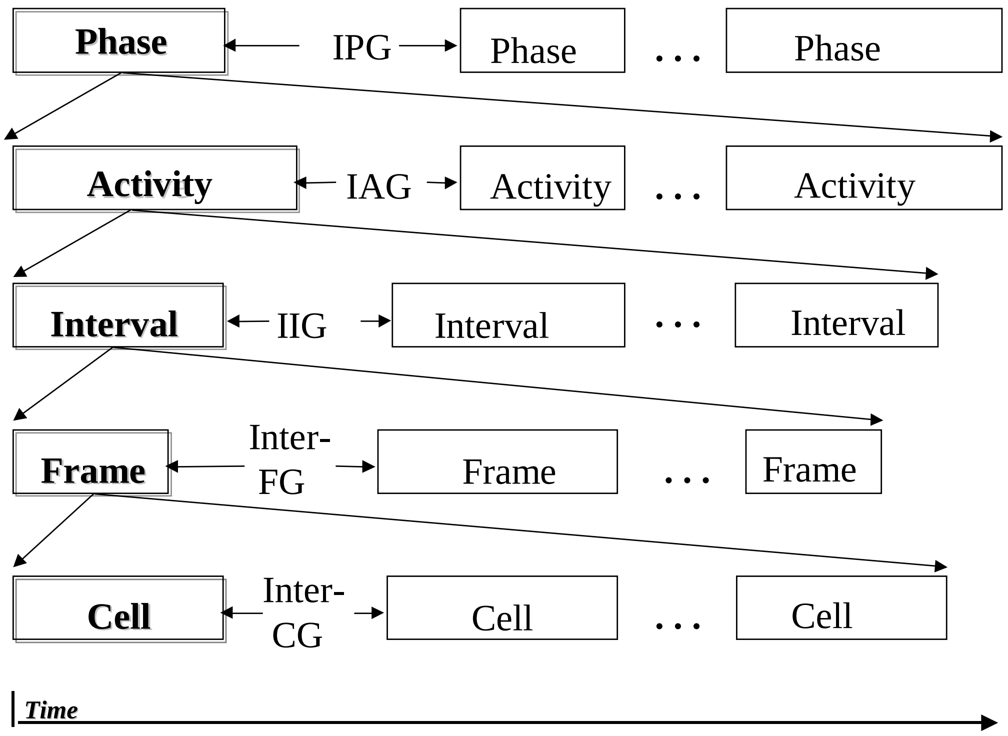


Figure 3.4: Relationship between Frame RLM source parameters

1. A test consists of one or more test phases. *E.g.*, phases 1 and 2 might represent a test with one and ten traffic sources.
2. Each test phase (except the last) is followed by an inter-phase gap during which no traffic is generated.
3. A test phase consists of one or more periods of activity. *E.g.*, a series of active periods might correspond to a series of traffic bursts.
4. Each active period (except the last) is followed by an inter-activity gap (idle period) during which no traffic is generated.
5. An active period consists of one or more characteristic intervals. *E.g.*, intervals might correspond to the bin widths used by a self-similar traffic generator.
6. Each characteristic interval (except the last) is followed by an inter-interval gap during which no traffic is generated.
7. Traffic characteristics typically vary from one interval to the next. *E.g.*, inter-frame gaps might vary among intervals, so that the inter-frame gap ~ exponential(10) in interval N and ~ exponential(15) in interval N+1.
8. Traffic characteristics typically do not vary within a characteristic interval. “Do not vary” does not imply that parameter values are constant. *E.g.*, the length of the inter-frame gap during interval N is exponentially distributed with mean 10. However, parameter values are random variates drawn from this distribution and are not constant within interval N.
9. A characteristic interval consists of one or more frames.
10. Each frame (except the last) is followed by an inter-frame gap during which no traffic is generated.
11. A frame consists of one or more cells.

12. Each cell (except the last) is followed by an inter-cell gap during which no traffic is generated.
13. Gaps at any level can be of length zero. *E.g.*, phases 1 and 2 might represent a test with one and ten traffic sources, and with no pause during the transition.

3.3.3. Multiplexer Parameters

There is one parameter of interest to define the behavior of the multiplexer.

> *Arbitration Algorithm* - Algorithm used to arbitrate amongst the various frame sources. This algorithm must be a well-defined procedure that specifies which of the various sources with a cell to transmit, get to transmit a cell at any time. Included are any parameters that are required by the arbitration algorithm.

The author of the frame reference load model will define the arbitration scheme for the multiplexer device. The specification of this arbitration scheme must fully specify how all sources will be multiplexed so that the traffic pattern for any RLM is well defined at the cell-level.

The service rate of the arbitration device is assumed to be the output link rate, therefore idles occur in a RLM if and only if there are no sources with cells to send.

3.3.4. Definition of a Frame RLM

To properly define a frame RLM, the following must be specified:

1. All frame sources must be defined and numbered.

2. The multiplexer arbitration scheme must be specified to arbitrate between the sources.

To specify a frame source, the frame source parameters are defined as being distributed with well-defined distributions. These can be any mathematical, statistical or algorithmic distribution including constant. With proper definition of distributions for these parameters, a variety of useful frame sources can be constructed.

Section 3.3.5 and Appendix E contain example Frame RLMs.

To fully specify the multiplexer operation, the arbitration algorithm for the multiplexer must be stated.

Definition of these parameters specifies a well-defined RLM. It should be noted that these RLMs might be statistical in nature. If any of the parameters are based on statistical distributions, then the cell-level traffic patterns are statistically reproducible over time.

3.3.5. Example of the Definition of a Frame RLM

A hypothetical video compression frame stream could be defined to have the following characteristics:

1. There are two video frame sources with the following properties.
   - A new data frame starts every 9 cell times.
   - The compressed frame size varies uniformly between 3 and 6 cells.

- The data frames burst at line rate. That is the ICG = 0.

2. The 2 sources are multiplexed with round robin arbitration.

Therefore, the frame RLM would be defined with the following parameters:

Source 0
FS = Uniform(3,6)
ICG = 0
$IFG_i = 9 - FS_i$ i = Frame Number
Source 1
FS = Uniform(3,6)
ICG = 0
$IFG_i = 9 - FS_i$ i = Frame Number
Arbitration; Round Robin

```
Number_Of_Sources = 2.
Round_Robin ( Number_Of_Sources ) {
    Current_Source = 0
    while ( True ) {
        if ( Cell_To_Transmit ( Current_Source ) ) {
            Send_Cell ( Current_Source )
        }
        Current_Source = ( Current_Source + 1) mod Number_Of_Sources
    }
}
```

The following diagram illustrates this example.

Example: Compressed video stream
Source 0
FS = Uniform(3,6)
ICG = 0
$IFG_i = 9 - FS_i$ i = Frame Number

Source 1
FS = Uniform(3,6)
ICG = 0
$IFG_i = 9 - FS_i$ i = Frame Number

Arbitration: Round Robin

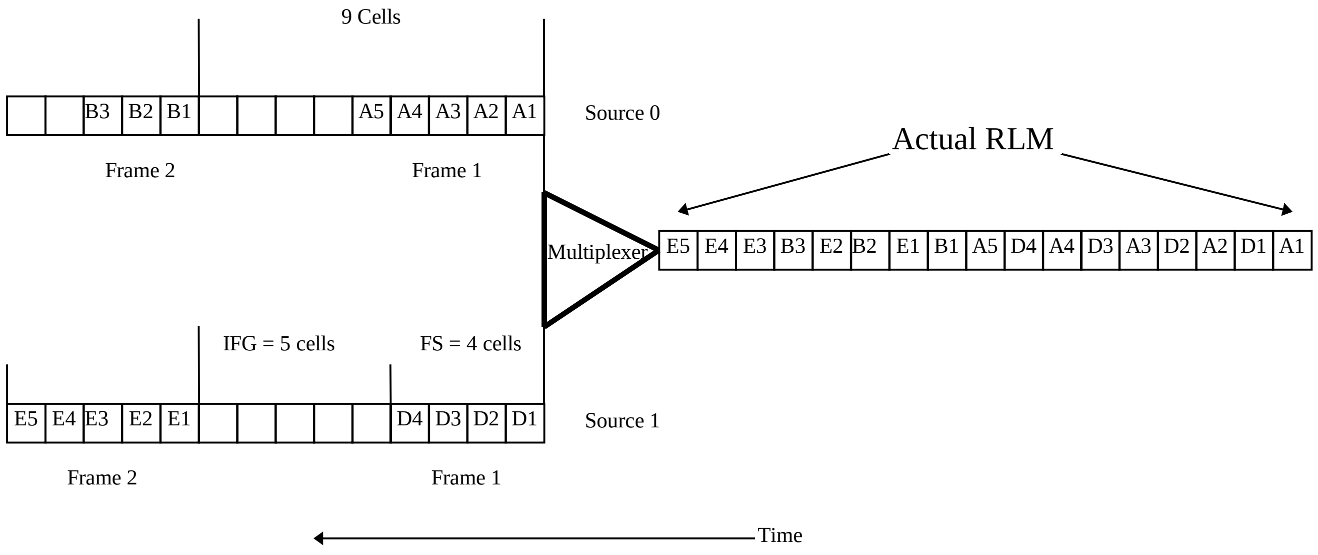


Figure 3.5: Example Frame Reference Load Model of Two Video Sources

3.4. Test Configurations

Each Reference Load (RL) defined in Section 3.3 must be verifiable at the measurement point MP1, defined in Section 3.2. This specification does not (and should not) mandate any particular method for generating a RL. Some test configurations may include connections that are routed through the IUT several times by looping links (as a way to produce a traffic load on a large number of IUT input interfaces while using a small number of traffic generators – see Appendix B). In such cases, since the behavior of the IUT is formally unknown, so is the exact nature of the aggregate traffic passing through it. Therefore, the traffic pattern is unverifiable at each subsequent ingress to the IUT, as some of the input traffic comes from the IUT outputs.

Consider the following hypothetical example: we might attempt to measure the delay characteristics of two IUTs at an input port load of 98%. One IUT exhibits a CLR of 10^{-5} while the other 0.05. The load at the inputs of the IUT with the higher loss might be around 93% or lower while the load on the inputs of the other IUT would be much closer to the intended 98%.

Since the behavior of the two IUTs is different, there is a discrepancy between the test conditions under which the two IUTs are being observed.

The formal discrepancy of an unverifiable RL may be avoided by ensuring that those components of a test configuration that introduce dependencies on IUT performance are included in the definition of the system being tested. Therefore, should the test configuration incorporate looping configurations, the SUT should be defined to include the looping links.

This approach has the advantage of being consistent with the definitions of scaleable test configurations in Appendix B. It is the responsibility of the tester to verify that the test configuration does indeed produce the specified RL (for example by moving a cell stream analyzer around to each of the inputs and repeating the test).

A system with n ports can be tested for the following connection configurations (for connections internal to the SUT):

- n-to-n straight,
- n-to-(n–1) full cross,
- n-to-m partial cross, $1 \le m \le n-1$,
- k-to-1, 1<k<n,
- 1-to-(n–1) multicast,
- n-to-(n–1) multicast.

Different connection configurations are illustrated in Figure 3.6, where each configuration includes one ATM switch with four ports, with their input components shown on the left and their output components shown the right.

In the case of n-to-n straight, input from one port exits to another port. This represents almost no path interference among the *n* VCCs. See Figure 3.6a.

In the case of n-to-(n–1) full cross, input from each port is divided equally to exit on each of the other (*n*–1) ports. This represents intense competition for the switching fabric by the $n \times (n-1)$ VCCs. See Figure 3.6b.

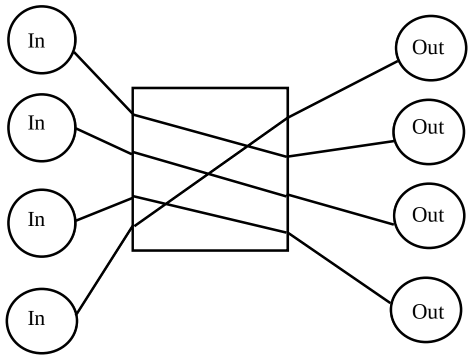
In the case of n-to-m partial cross, input from each port is divided equally to exit on the other *m* ports ($1 \le m \le n-1$). This represents partial competition for the switching fabrics by the $n \times m$ VCCs as shown in Figure 3.6c. Note that n-to-n straight and n-to-(n–1) full cross are special cases of n-to-m partial cross with m=1 and m=n–1, respectively.

In the case of k-to-1, input from *k* ($1 < k < n$) ports is destined to one output port. This stresses the output port logic. There are *k* VCCs as shown in Figure 3.6d.

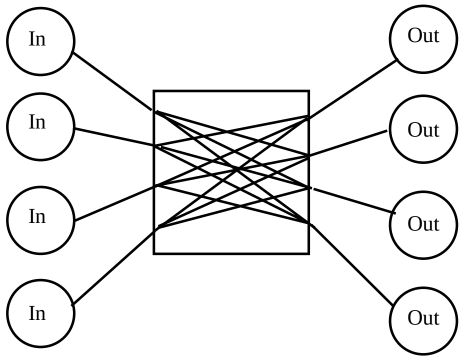
In the case of 1-to-(n–1) multicast, all frames input on the one designated port are multicast to all other (n–1) ports. This tests single multicast performance of the switch. There is only one (multicast) VCC as shown in Figure 3.6e.

In the case of n-to-(n–1) multicast, input from each port is multicast to all other (*n*–1) ports. This tests multiple multicast performance of the switch. There are *n* (multicast) VCCs. See Figure 3.6f.

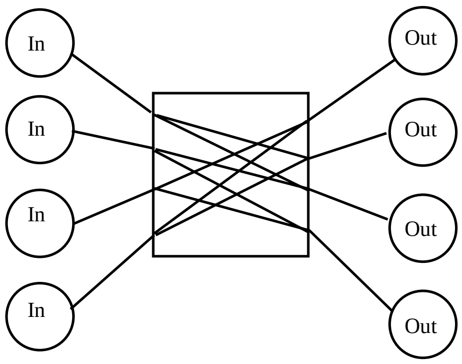
Note that a generalization of 1-to-(n-1) multicast and n-to-(n-1) multicast is m-to-(n-1) multicast with $1 \le m \le n$.



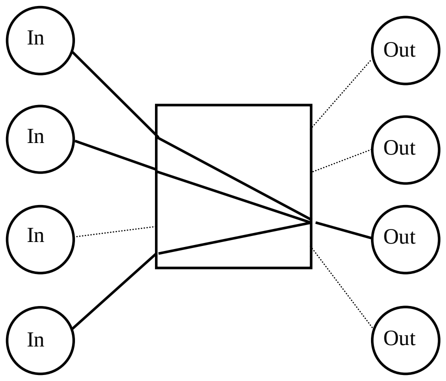
a. n-to-n straight: *n* VCCs; *n*=4



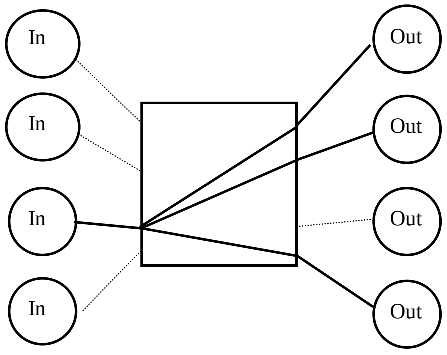
b. n-to-(n–1) full cross: *n*× (*n*–1) VCCs; *n*=4



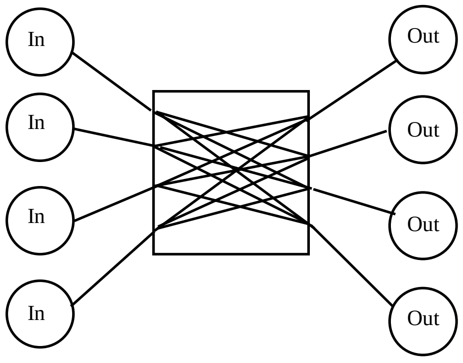
c. n-to-m partial cross: *n*×*m* VCCs; *n*=4, *m*=2



d. k-to-1: *k* VCCs; *k*=3



e. 1-to-(n-1) multicast: one (multicast) VCC



f. n-to-(n–1) multicast: *n* (*multicast*) VCCs; *n*=4

Figure 3.6: Connection Configurations for Foreground Traffic

3.4.1. Foreground Traffic

Before starting measurements, a number of VCCs (or VPCs), henceforth referred to as "foreground VCCs," are established through the SUT. Foreground VCCs are used to transfer only the traffic whose performance is measured. That traffic is referred to as the foreground traffic.

Foreground traffic is specified by the type of foreground VCCs, connection configuration, service category, arrival patterns, frame length and input rate.

Foreground VCCs can be permanent or switched, virtual path or virtual channel connections, established between ports on the same network module on the switch, or between ports on different network modules, or between ports on different switching fabrics.

The list of foreground traffic characteristics and their possible values are now provided:

- type of foreground VCCs: permanent virtual path connections, switched virtual path connections, **permanent virtual channel connections**, switched virtual channel connections;
- foreground VCCs established: between ports inside a network module, **between ports on different network modules**, between ports on different fabrics, some combination of previous cases;
- connection configuration: one of the variants shown in Figure 3.6, e.g., **n-to-m partial cross**;
- service category: CBR, **UBR**, ABR, rt-VBR, nrt-VBR, and GFR (GFR is a frame-aware service category that does not apply to virtual path connections);
- frame size: 64, **1518**, **9188,** 65535 bytes or variable;
- inter-frame gap: constant, random;
- inter-cell gap: constant, random.

Values in bold indicate default traffic characteristics that, unless stated otherwise for a specific metric, it is recommended be used in performance testing.

The maximum foreground load (MFL) is defined as the sum of all physical link rates used for transmission of the foreground traffic, in the current switch configuration. Input rate of the foreground traffic is expressed in the effective bits/sec, counting only bits from frames, excluding the overhead introduced by the ATM technology and transmission systems.

3.4.2. Background Traffic

In connection configurations with multiple VCCs, it is possible to use some VCCs for foreground traffic and the others for background traffic. One particularly interesting case is when one VCC of the n-to-n straight configuration is used for foreground and the remaining n-1 VCCs are used for background. This will help study the effect of background traffic on the quality of service of the foreground traffic.

Background traffic characteristics that affect performance are the type of background VCCs, connection configuration, service category, arrival patterns (if applicable), frame length (if applicable) and input rate.

Like the foreground VCC, background VCCs can be permanent or switched, virtual path or channel connections, established between ports on the same network module on the switch, or between ports on different network modules, or between ports on different switching fabrics. To avoid interference on the traffic generator/analyzer equipment, background VCCs are established in such a way that they do not use the input link or the output link of the foreground VCC in the same direction.

For an SUT with *n* ports, the background traffic can use (*n*–2) ports, not used by the foreground traffic, for both input and output. The port with the input link of the foreground traffic can be used as an output port for the background traffic. Similarly, the output port of the foreground traffic can be used as an input port for the background traffic. Overall, background traffic can use an equivalent of *w*=*n*–1 ports. The maximum background load (MBL) is defined as the sum of all physical link rates, except the one used as the input link for the foreground traffic, in the current switch configuration.

For background traffic, an SUT with *n* (=*w*+1) ports will support any one of the following background traffic connection configurations:

- w-to-w straight, with *w* background VCCs, (Figure 3.6a);
- w-to-(w–1) full cross, with *w*×(*w*–1) background VCCs. (Figure 3.6b);
- **w-to-m partial cross**, 1 ≤ *m* ≤ *w*–1, with *w*×*m* background VCCs. (Figure 3.6c);
- 1-to-(w–1) multicast, with one (multicast) background VCC. (Figure 3.6e);
- n-to-(w–1) multicast, with *w* (multicast) background VCC. (Figure 3.6d).

The list of background traffic characteristics and their possible values are now provided:

- type of background VCCs: permanent virtual path connections, switched virtual path connections, **permanent virtual channel connections**, switch virtual channel connections;
- background VCCs established: between ports inside a network module, **between ports on different network modules**, between ports on different fabrics, some combination of previous cases;
- service category:
  - **UBR**: priority equal to or lower than that of the foreground traffic;
  - **CBR**: priority equal to or higher than that of the foreground traffic;
  - rt-VBR, nrt-VBR, ABR, and GFR service categories are under study;
- arrival patterns: **equally spaced frames**, self-similar, random;
- frame size: 64, 1518, **9188**, 65535 bytes or variable;
- inter-frame gap: constant, random;
- inter-cell gap: constant, random;
- input rate: **0**, 0.5, 0.75, **0.875**, 0.9375, 0.9687, … (*i.e.*, $1 - 2^{-k}$, k = 0, 1, 2, 3, 4, 5,…) of MBL.

Values in bold indicate default traffic characteristics that, unless stated otherwise for a specific metric, it is recommended be used in performance testing. Sometimes background traffic is not required (represented here by an input rate of “0”). When background traffic is required, the default input rate should be 0.875 of MBL. The UBR service category should be used if it is intended for the foreground traffic to have priority over the background traffic, and the CBR service category should be used for the inverse scenario.

Input rate of the background traffic is expressed in the effective bits/sec, counting only bits from frames excluding the overhead introduced by the ATM technology and transmission systems.

3.5. General Measurement Procedures

Before conducting performance tests, it is recommended that the port clocks are synchronized or locked together; otherwise, unstable results may be observed. In case of instability, one solution is to reduce the maximum load placed on the IUT to slightly below 100%. In this case, the load used should be reported.

Performance tests can be conducted under two conditions, remembering that the performance metric of interest is recorded only for the foreground traffic:

- without background traffic, or
- with background traffic.

The procedure to measure the performance metric in this case includes a number of test runs. A test run starts with the traffic being sent at a given input rate over the foreground VCCs. The average frame latency is constantly monitored. A test run ends and the foreground traffic is stopped when the average frame latency has not significantly changed (not more than 5%) during a period of at least 5 minutes. This ensures stability of the IUT.

3.6. Statistical Variations

For the given foreground and background traffic, performance metric results are recorded for p frames, according to the procedures described in Section 3.5. Here p is a parameter and its default is 100.

Let R_i be the result of the $\underline{i}$th test. The sample mean, variance, standard deviation, and standard error of the mean of the measurement are computed as follows:

$$\text{Mean} = (\Sigma\, R_i) / p$$

$$\text{Variance} = (\Sigma(R_i - \text{Mean})^2) / (p-1)$$

$$\text{Standard deviation} = (\text{Variance})^{1/2}$$

$$\text{Standard error} = (\text{Variance} / p)^{1/2}$$

Depending on the input traffic, the sample size p=100 may not be large enough to use the asymptotic technique or the measured data may be correlated. If the measurements of R_i for all i are statistically independent, an unbiased estimate of the variance of the results is given above. However, in most measurement situations of practical interest, the measured results will not be statistically independent. In this case, techniques described in Appendix C.3 can be used to obtain an unbiased estimate of the variance of the results. Appendix C provides techniques for estimating confidence intervals. Appendix G provides simple recipes for standard statistical methods.

In some testing situations, it might be useful to compute and use statistics as soon as a measurement is obtained, rather than to compute them at the end of the test. The following recursive definition can be used to update statistics after each measurement:

```
measure $R_1$
$mean_1 = R_1$          /* statistic from iteration 1 available for use */
$variance_1 = 0$   /* $variance_1$ is undefined; this statement is used to simplify the computation of $variance_2$ */
for i = 2 to p {
        *measure* $R_i$
        $mean_i = mean_{i-1} + ((R_i - mean_{i-1}) / i)$
        $variance_i = ((1 - (1 / (i - 1))) \times variance_{i-1}) + (i \times (mean_i - mean_{i-1})^2)$
        $standard\_deviation_i = (variance_i)^{1/2}$
        $standard\_error_i = (variance_i / i)^{½}$
        /* statistics from iteration i available for use */
    }
```

3.7. Optional Traffic Management Functions and Procedures

The tester shall decide which subset of the available optional traffic management functions and procedures (from those listed in Section 5 of [2], or elsewhere) is to be implemented. Once the choice has been made, it shall be implemented consistently for the duration of the test, and reported along with the test results.

3.8. Reporting Results

A detailed description of the test configuration, including the SUT and RL, should always be attached to the test report. It should include details such as the number of ports, rate of each port, number of ports per network module, number of network modules, number of network modules per fabric, number of fabrics, maximum foreground load (MFL), maximum background load (MBL), software version, test equipment used, any optional performance enhancing schemes, traffic management functions and procedures that were implemented, and any other relevant information.

Values for the performance metric of interest, with corresponding input load are reported along with foreground (and background, if any) traffic characteristics. Appendix F provides two tables for reporting test results.

4. Performance Metrics

4.1. Throughput

4.1.1. Definitions

There are three frame-level throughput metrics of interest to a user:

- **Loss-less throughput** - It is the maximum rate at which none of the offered frames is dropped by the SUT.
- **Peak throughput** - It is the maximum rate at which the SUT operates regardless of frames dropped. The maximum rate can actually occur when the loss is not zero.
- **Full-load throughput** - It is the rate at which the SUT operates when the input links are loaded at 100% of their capacity.

An example of a graph of throughput *vs.* input rate is shown in Figure 4.1. Level X defines the loss-less throughput, level Y defines the peak throughput, and level Z defines the full-load throughput.

The loss-less throughput is the highest load at which the count of the output frames equals the count of the input frames. The peak throughput is the maximum throughput that can be achieved in spite of the losses. The full-load throughput is the throughput of the system at 100% load on input links. Note that the peak throughput may equal the loss-less throughput in some cases.

Only frames that are received completely without errors are included in frame-level throughput computation. Partial frames and frames with CRC errors are not included.

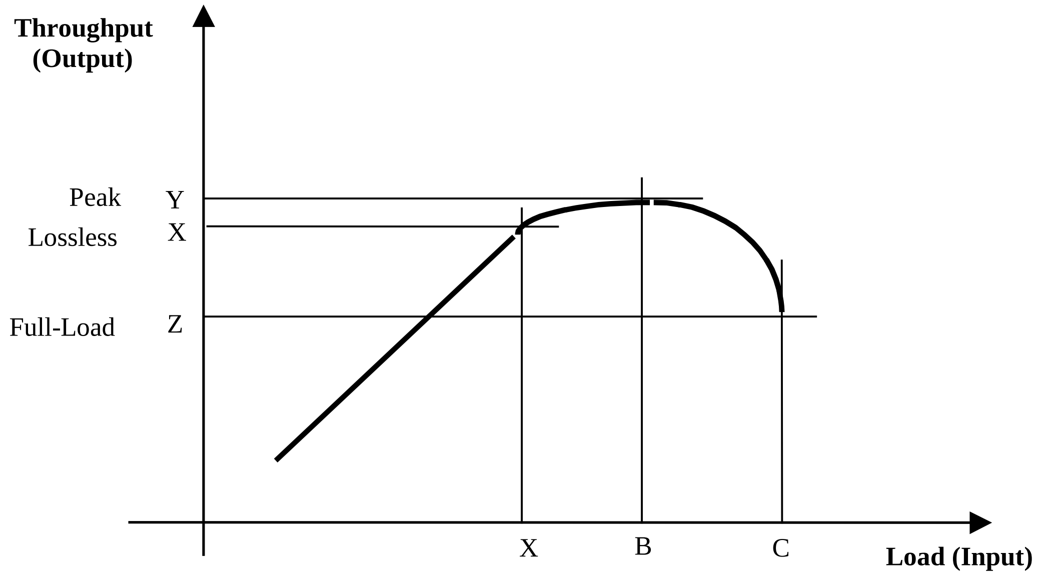


Figure 4.1: Peak, Loss-Less and Full-Load Throughput

4.1.2. Units

The preferred measurement for throughput is bits/sec. Bits/sec includes any overhead introduced by the SUT. Since for processing, the SUT may add headers and other information to the received data stream, its performance may be affected by the additional overhead it introduces. Also, for measurement purposes, it may not always be possible to identify and exclude overhead at the bit-level.

Alternately, frames/sec may be used, providing the frame size is specified in the results. Whichever measurement is used, it must be reported in the test results. Cells/sec, however, is not a good unit for frame-level performance, since the user normally doesn't see the cells.

4.1.3. Measurement Procedures

During a given test run period, several test runs will need to be executed, and the total number of frames sent to the SUT and the total number of frames received from the SUT are recorded. The throughput (output rate) is computed based on the duration of a test run and the number of received frames.

If the input frame count and the output frame count are the same, then the input rate is increased, and the test is conducted again.

The loss-less throughput is the highest throughput at which the count of the output frames equals the count of the input frames.

The input rate is then increased even further. Although some frames will be lost, the throughput may increase till it reaches the peak throughput value. After this point, any further increase in the input rate will result in a decrease in the throughput.

The input rate is finally increased to 100% of the input link rates and the full-load throughput is recorded.

In the best case, where the frames are of fixed size and equally spaced, and the load is increased in steps, there is always measurement inaccuracy of at least the step size. If the frame size and spacing can vary, then the results can vary from one measurement to the next. Appendix C provides techniques for estimating confidence intervals for statistical parameters of interest, such as the means and/or variances of throughputs.

4.2. Frame Latency

Two of the frame latency metrics of interest are First In Last Out (FILO), and Message-In Message-Out (MIMO). FILO indicates user perceived performance, whereas MIMO indicates latency that is introduced inherently by the SUT. Both are explained in Appendix A. Appendix A.12 also indicates other frame latency metrics of interest for additivity and end-to-end frame latency.

4.2.1. Definition

The previously defined frame reference events FRE1 and FRE2 and successful frame transfer outcome can be used to define a performance parameter for characterizing the information delay of the frame transfer outcome.

The FILO frame transmission delay parameter is defined as the time T2 of a FRE2 at MP2 minus the time T1 of a corresponding FRE1 at MP1, where the corresponding FRE2 and FRE1 are related as a successful frame transfer outcome. FILO is the information delay performance parameter that is directly measured. Other definitions of frame latency (*e.g.*, MIMO) may be derived.

This definition excludes those FRE1 and FRE2 events associated with Corrupted Frame Transfer Outcomes, because such outcomes can be less reflective of normal user information delays.

FILO latency is defined as follows:

FILO latency = Time between the first-bit entry and the last-bit exit

MIMO latency is defined as follows:

$$\text{MIMO latency} = \text{FILO latency} - \text{FILO}_0$$

where FILO_0 (Nominal Frame Output Time) is defined as:

FILO_0 = Time a frame needs to pass through the *zero-delay switch*

FILO_0 can be calculated using the following procedure:

Initially $\text{FILO}_0 = 0$ and time t is measured from the arrival of the first bit of the first cell. For each cell with its first bit arriving at time t $\Rightarrow$ $\text{FILO}_0 = \max\{t, \text{FILO}_0\} + \text{CT}$.

Here CT is the larger of the cell input time or cell output time. Cell times are computed as the cell size of 424 bits divided by the respective link rates in bits per second.

An equivalent MIMO latency definition is:

$$\text{MIMO latency} = \begin{cases} \text{LILO latency} & \text{if Input Link Rate} \leq \text{Output Link Rate} \\ \text{FILO latency} - \text{FILO}_0 & \text{otherwise} \end{cases}$$

where

- LILO latency = Time between the last-bit entry and the last-bit exit

Frame Latency Measurements and Calculation

To obtain latency for a given frame, the time of occurrence for the following two events need to be recorded:

- First-bit of the frame enters into the SUT, and
- Last-bit of the frame exits from the SUT.

The time between the second and the first events is FILO latency. If measurement data are available at cell-level, it can be shown that:

FILO latency = First cell's transfer delay + First cell to last cell inter-arrival time

where

- cell transfer delay (CTD) is the time between the first bit of the cell entering the switch and the last bit of the cell leaving the switch,
- cell inter-arrival time is the time between arrival from the switch of the last bit of the first cell and arrival from the switch of the last bit of the second cell.

Given the cell pattern of a frame on input, FILO_0 can be obtained using the procedure from its definition. Then, substituting FILO latency and FILO_0 in the MIMO latency formula would give the SUT delay for the given frame.

In the cases when Input Link Rate ≤ Output Link Rate, MIMO latency can be obtained easier. In those cases, the time of occurrence for the following two events need to be recorded:

- Last-bit of the frame enters into the SUT,
- Last-bit of the frame exits from the SUT.

The time between the second and the first events is LILO latency. When measurement data are available at cell-level, it can be shown that:

LILO latency = Last cell's transfer delay – Cell input time

and in these cases, LILO latency would give the SUT delay for the given frame.

An explanation of MIMO latency and its justification is presented in Appendix A.

4.2.2. Units

The latency should be specified in seconds.

4.2.3. Measurement Procedures

For frame latency measurements, it is first necessary to establish one VCC (or VPC) used only by foreground traffic, and a number of VCCs or VPCs used only by background traffic. Then, the background traffic is generated. Characteristics of background traffic are described in Section 3.4.2. When flow of the background traffic has been established, the foreground traffic is generated. Characteristics of foreground traffic are specified in Section 3.4.1. After the steady state flow of foreground traffic has been reached the required times and/or delays needed for frame latency calculation are recorded for *p* consecutive frames from the foreground traffic, while the flow of background traffic continue uninterrupted. The entire procedure is referred to as one measurement run. It is recommended that frame latency be measured for at least the range of fixed frame lengths listed in Section 3.4.1. The use of variable length frames is under study.

The first measurement run is performed at the lowest possible foreground input rate (for the given test equipment). For later measurement runs, the foreground load is increased up to the point when losses in the traffic occur or up to the full foreground load (FFL). FFL is equal to the lesser of the input and the output link rates used by the foreground VCC. Suggested input rates for the foreground traffic are: 0.5, 0.75, 0.875, 0.9375, 0.9687, ..., *i.e.*, $1 – 2^{-k}$, k = 1, 2, 3, 4, 5, …, of FFL.

4.2.4. Burst Model Measurement Procedures

Traffic sources often generate bursty traffic at the frame-level. For example, a burst may occur when a data file is to be transmitted, but is too large to be encapsulated in a single frame due to MTU constraints. Based on the RLM framework in Section 3.3, MIMO latency can be used to measure the latency required for a device to handle bursts of frame based traffic. For instance, rt-VBR is a delay sensitive service that is defined to handle traffic within the burst tolerance (see ATM Forum Traffic Management Specification 4.1 [2]). The burst tolerance of the rt-VBR service will dictate whether or not QoS commitments apply to the burst, and the MIMO latency measured for the burst can be used to indicate whether or not the QoS commitments on delay are met. The latency performance for the non-real time service categories is not part of the specified QoS, but may be of interest.

When frame bursts are generated by encapsulating user data in multiple frames and transmitting them at access rate, better MIMO latency, measured at the frame burst level, contributes directly to performance perceived by the end-user. For instance the latency induced on a connection is important for voice applications and some video applications.

The service categories CBR, rt-VBR, nrt-VBR, and GFR use traffic parameters that may, under specific conditions, not allow QoS commitments to be given to a bursty RLM. In order for QoS commitments to apply, the traffic parameters must be adequately specified to handle the burst characteristics of the source. For instance, the PCR, MCR, SCR, MaxCTD and MBS must be large enough to handle the burst of back to back cells that results from the burst of frames. The exact specification of these values depends on the SUT, connection rate and the input frame RLM (as defined in Section 3.3.2). It is inappropriate to measure MIMO latency for a burst that does not satisfy the traffic parameters of the established connections. For the service categories ABR and UBR, the bursty frame RLM must be conformant only to the PCR traffic parameter (with its associated CDVT) to be QoS eligible, as no QoS commitments to delay, burst tolerance or sustainable rate are given.

The following frame reference load model, based on Section 3.3.2, can be used to measure the latency in the SUT for bursty traffic:

1. The number of frames (N) and the frame size (S) are positive integers and must be defined such that $N \times S$ = Burst Size. The burst size for which MIMO latency may be measured

ranges from very small (*e.g.,* 0.1 kbytes of user data) to the maximum that may be supported by the negotiated traffic parameters of the connection.

2. Frame sizes are constant. Within a frame, should multiple cells be needed to meet the desired frame size (S), a constant inter-cell gap should be chosen so that PCR is maintained. The specified CDVT should be large enough to handle the generated traffic.
3. Should multiple frames be needed (N>1) to meet the desired burst size, they must be sent separated by a constant inter-frame gap. For VBR connections, the size of the gap determines the SCR, and for all other service categories, the inter-frame gap should be set so that PCR is maintained. Where appropriate, the specified CDVTs, MBS and MFS should be large enough to handle the generated traffic.
4. For repeated measurement of the MIMO latency for frame bursts, repeated bursts may be sent, separated by an inter-interval gap that is chosen to be large enough so that the negotiated traffic parameters of the connection are not violated.

The latency measurement should be taken after sufficient time has passed to allow the connections to be beyond any initial settling period.

It may also be of interest to measure the throughput for the above traffic sources concurrently. Good latency performance does not necessarily lead to good throughput performance and vice versa.

4.3. Fairness Index

4.3.1. Definition

Given *n* virtual circuits sharing a system (a single switch or a network of switches) and contending for the resources, fairness index indicates how far the actual individual allocations are from the ideal allocations. Fairness index can be applied to several metrics, such as throughput and latency. In the simplest case for a total throughput *T*, the ideal allocation should be *T/n*. However, other fairness criteria may be used, such as those specified in Appendix I.3 of TM 4.1 [2] for the ABR service category.

If the actual measured throughputs of *n* virtual circuits are found to be $\{T_1, T_2, ..., T_n\}$, where the ideal throughputs should be $\{\hat{T}_1, \hat{T}_2, ..., \hat{T}_n\}$, then the throughput fairness of the system under test is quantified by the "fairness index" computed as follows:

$$\text{Fairness index} = (\sum x_i)^2 / (n \times \sum x_i^2)$$

where:

- $x_i = T_i/\hat{T}_i$ is the relative allocation to i^{th} VC.

There are two throughput fairness metrics that are of interest to users:

- *Peak throughput fairness*: This is the fairness at a frame load for the peak throughput.
- *Full-load throughput fairness*: This is the fairness at a frame load for the full-load throughput.

In the case of Latency, $\{T_1, T_2, ..., T_n\}$ represents the actual measured latency of *n* virtual circuits, while $\{\hat{T}_1, \hat{T}_2, ..., \hat{T}_n\}$ should be the ideal latency ($FILO_0$). The latency fairness of the system under test is quantified by the above defined "fairness index". However, extreme unfairness in latency is expected to usually show up as unfairness in throughput and *vice-versa*. Therefore, it is not required to quantify fairness of latency.

4.3.2. Units

This fairness index is dimension-less. The units of measurements made to calculate the fairness index (bits/sec, cells/sec, or frames/sec) do not affect its value. In addition, the fairness index has the following desirable properties:

- It is a normalized measure that ranges between zero and one. The maximum fairness is 100% and the minimum 0%. This makes it intuitive to interpret and present.
- If all x_i's are equal, the allocation is fair and the fairness index is one.
- If n-k of n x_i's are zero, while the remaining k x_i's are equal and non-zero, the fairness index is k/n. Thus, a system which allocates all its capacity to 80% of VCs has a fairness index of 0.8 and so on.

4.3.3. Measurement Procedures

The following are examples for applying the fairness index to the throughput metric.

To measure a peak throughput fairness, the peak throughput for the given SUT must be first obtained as described in Section 4.1.3. An experiment for peak throughput fairness is performed by generating the input load corresponding to the peak throughput for time (t) and recording throughput for each foreground virtual circuit. The experiment is repeated p times. Here t is expressed in seconds, and p is a parameter whose default value is 30.

Similarly, to measure a full-load throughput fairness index, the full-load throughput for the given SUT must be obtained first as described in Section 4.1.3.

4.4. Frame Loss Ratio

4.4.1. Definition

Frame Loss Ratio is defined as the fraction of frames that are corrupted or not forwarded by a System Under Test (SUT) due to several reasons, such as lack of resources.

Following the approach adopted in the definition of other metrics in this specification two frame transfer outcomes are first proposed, and are then used to define a Frame Loss Ratio (FLR). This Frame Loss Ratio is proposed as a performance parameter for characterizing the information loss performance of frame transfer outcome.

Using the definitions for frame-based Reference Events, two frame transfer outcomes are defined:

1. Successful Frame Transfer Outcome is defined as the occurrence at MP2 of a FRE2 corresponding to the occurrence at MP1 of a FRE1, within a time T_{max} to be specified, if each user information bit in the FRE2 is identical to the corresponding user information bit in the corresponding FRE1.
2. Corrupted Frame Transfer Outcome is defined as being either

(a) the lack of occurrence at the MP2 of a FRE2 corresponding to the occurrence at MP1 of a FRE1, within a time T_{max} to be specified, where each user information bit in the FRE2 must be identical to the corresponding user information bit in the corresponding FRE1, or
(b) the occurrence at MP2 of a FRE2 for which there is no corresponding FRE1.

Use of a limit, T_{max}, on the maximum permissible frame transfer time classifies unduly delayed frames as corrupted frame transfer outcomes. The possible occurrence of a "lost" frame is included in the above definition as a Corrupted Frame Transfer Outcome. The possible occurrence of a "misinserted" frame is included in the above definition as corrupted frame transfer outcome.

This definition of corrupted frame transfer outcome does not distinguish between the failure of a FRE2 to match bit-for-bit with the user information in the corresponding FRE1 due to the supporting ATM connection's experiencing a lost cell, errored cell, or misinserted cell. Regardless of the underlying impairment cause at the supporting ATM layer, this resulting frame is unlikely to be usable by most higher layer applications and therefore contributes to information loss for this frame transfer outcome.

The Frame Loss Ratio is defined as:

$$FLR = \frac{Corrupted\ Frame\ Transfer\ Outcomes}{Successful\ Frame\ Transfer\ Outcomes + Corrupted\ Frame\ Transfer\ Outcomes}$$

where the Successful Frame Transfer Outcomes and Corrupted Frame Transfer Outcomes belong to some population of interest. This population of interest could, for example, contain all such frame transfer outcomes that occur between a specific pair of MPs during a stipulated time interval.

There are two frame loss ratio metrics that are of interest to a user:

- *Peak throughput frame loss ratio*: This is the frame loss ratio at a frame load for the peak throughput.
- *Full-load throughput frame loss ratio*: This is the frame loss ratio at a frame load for the full-load throughput.

The Frame Loss Ratio for point-to-multipoint, multipoint-to-point and multipoint-to-multipoint connections is for further study.

An alternative metric, called Application Goodput (AG) can be defined as:

Application Goodput = Frames Received / Frames Transmitted

This metric captures the notion of what an application sees as useful data transmission in the long term. At the AAL-level, AG can be calculated as follows:

AG = 1 - FLR

4.4.2. Units

The frame loss ratio is expressed as a fraction of input frames.

4.4.3. Measurement Procedures

The frame loss ratio metric is related to the throughput:

Frame Loss Ratio = (Input Rate - Throughput)/Input Rate

Thus, no additional experiments are required for frame loss ratios. These can be derived from tests performed for throughput measurements.

4.5. Maximum Frame Burst Size (MFBS)

4.5.1. Definition

Maximum Frame Burst Size (MFBS) is the maximum number of frames that each of source end systems can send at the peak rate through a system under test without incurring any loss. MFBS measures the data buffering capability of the SUT and its ability to handle back-to-back frames.

Many applications and transport layer protocol drivers often present a burst of frames to AAL for transmission. For such applications, Maximum Frame Burst Size provides a useful indication.

This metric is particularly relevant to UBR service category since the UBR sources are always allowed to send a burst at peak rate. ABR sources may be throttled down to a lower rate if a switch runs out of buffering resources.

4.5.2. Units

MFBS should be expressed in octets of AAL payload field. This is preferred over number of frames or cells. The former requires specifying the frame size and the latter is not very meaningful for a frame-level metric. Also, number of cells has to be converted to octets for use by AAL users.

4.5.3. Measurement Procedures

The MFBS is measured for the k-to-1 connection configuration as shown in Figure 3.6. Thus, k VCCs (or VPCs) are established through the SUT.

The measurement procedure may require a number of tests. Each test includes simultaneous generations of fixed length bursts of back-to-back cells through all k VCCs (or VPCs) and counting of all cells transmitted by the SUT. If there is no loss of cells, the length of bursts is increased, but if there is a loss, the length of bursts is decreased. In both cases, the next test is performed with the new burst length. The procedure is finished when the maximum cell burst size (MCBS) is found. MCBS is the maximum burst length for which there is no cell loss.

Tests are conducted without any background traffic.

Given MCBS, one can calculate the maximum integral number of back-to-back frames of a given size, which can be sent into the SUT of the given connection configuration and delivered by the SUT without any loss. This integral number then converted to octets of AAL payload field to obtain the Maximum Frame Burst Size (MFBS).

There is no need for obtaining more than one sample for MFBS. Consequently, there is no need for calculation of the means and/or variances.

5. References

[1] af-test-0022.000 (Introduction to ATM Forum Test Specifications 1.0), 1994

[2] af-tm-0121.000 (Traffic Management Specification 4.1), 1999

[3] ITU-T Recommendation I.150 (1995), B-ISDN Asynchronous Transfer Mode Functional Characteristics

[4] ITU-T Recommendation I.350 (1993), General Aspects of Quality of Service and Network Performance in Digital Networks, Including ISDNs

[5] ITU-T Recommendation I.353 (1996), Reference events for defining ISDN and B-ISDN performance parameters

[6] ITU-T Recommendation I.356 (1996), B-ISDN ATM layer cell transfer performance

[7] B. Efron and B. Tibshirani, An Introduction to the Bootstrap, Chapman and Hall, 1993.

[8] ITU-T Recommendation I.361 (1995), B-ISDN ATM layer specification

[9] ITU-T Recommendation O.191 (1997), Equipment to assess ATM layer cell transfer performance

[10] IETF RFC1242-1991, Benchmarking Terminology for Network Interconnection Devices

[11] ITU-T Recommendation X.135 (1992), Speed of service (delay and throughput) performance values for public data networks when providing international packet-switched services

[12] ITU-T Recommendation I.610 (1995), B-ISDN operation and maintenance principles and functions

[13] ITU-T Recommendation X.144 (1995), User information transfer performance parameters for data networks providing international frame relay PVC service

[14] ITU-T Draft Recommendation X.144 Amendment 1, Annex C(1996), Some Relations Between Frame-level and ATM-level Performance Parameters, published in COM 7-13, 1997

[15] ITU-T Recommendation X.145 (1996), Performance for data networks providing international frame relay SVC service

[16] Computer Performance Modeling Handbook, S. Lavenberg (ed.), 1983

[17] IETF RFC 1242, Benchmarking Terminology for Network Interconnection Devices

Appendix A: Defining Frame Latency on ATM Networks

A.1. Introduction

This appendix discusses delays, and the performance metrics characterizing them, that an ATM network introduces to its frames. We are concerned with delays caused by node processing, such as switching and routing, as well as queuing delays that may be introduced by the background traffic and inter-network link transmission delays. On the other hand, transmission delays introduced by input and output links of a network component should not be attributed to the component. Also, note that characteristics of traffic generators (*e.g.*, host speeds) should not affect network performance metrics. The discussion in this Appendix applies to any network element (including switches, multiplexers, inverse-multiplexers, wires) or any combination of such network elements. Although we frequently use the term “switch,” the discussion applies equally well to other network elements, whole networks, or parts of networks.

In the case of a single bit, the switch (network) delay is generally defined as the time between the instant the bit enters the system and the instant the bit exits from the system. Figure A.1 illustrates the single-bit latency.

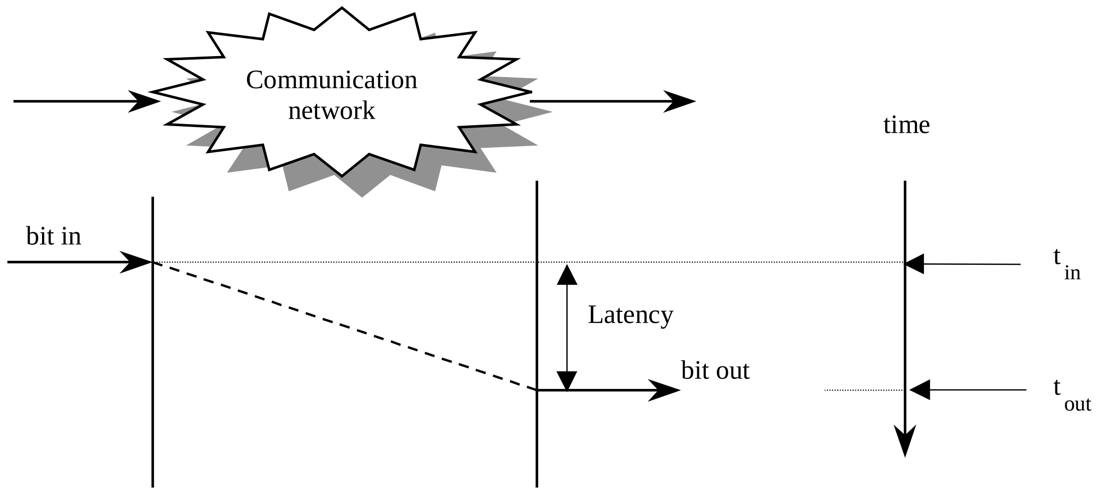


Figure A.1: Latency for a Single Bit

For multi-bit frames, the usual way to define the frame latency introduced by a switching device is to apply one of the following four definitions:

- FIFO latency: Time between the first-bit entry and the first-bit exit
- LILO latency: Time between the last-bit entry and the last-bit exit
- FILO latency: Time between the first-bit entry and the last-bit exit
- LIFO latency: Time between the last-bit entry and the first-bit exit

Figure A.2 illustrates the usual frame latencies (FIFO, LILO, FILO and LIFO) in a scenario with a contiguous frame on both input and output, passing through the given communication network which has an input link rate lower than the output link rate.

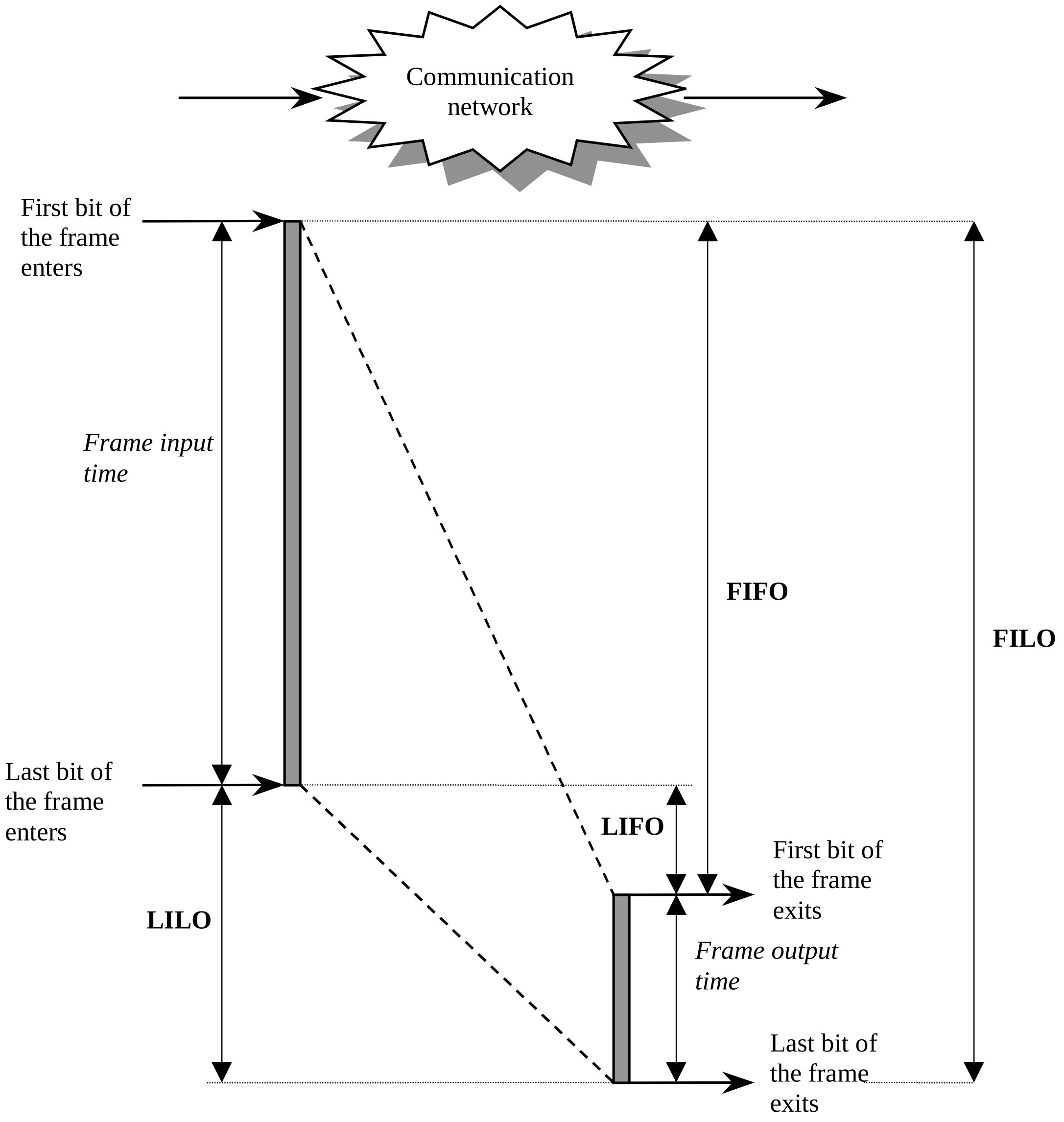

Communication network
First bit of the frame enters
Frame input time
Last bit of the frame enters
LILO
FIFO
FILO
LIFO
First bit of the frame exits
Frame output time
Last bit of the frame exits

Figure A.2: Usual Frame Latencies

Unfortunately, as it will be shown later, none of the four above metrics is appropriate for an equipment perspective of frame latency. In this appendix, we introduce and justify a new latency metric called "MIMO" latency. This new latency metric applies to any type of network where the frames may be contiguous or discontinuous, although our primary interest is an ATM environment. To define the MIMO latency, we introduce the concept of a "zero-delay" switch, which is in some sense the best a switch can do. The delay of any other switch is defined as the latency over and above the delay of a zero-delay switch.

This appendix is organized as follows. In the next section, we discuss the applicability of various latency metrics to the ATM environment. We introduce the MIMO latency in Appendix A.3. In Appendix A.4, we introduce the concept of a zero-delay switch and its processing of individual cells and contiguous frames. We discuss delays introduced to discontinuous frames passing through a zero-delay switch in Appendix A.5. Appendix A.6

presents the method for calculating the FILO latency of frames passing through a zero-delay switch. An equivalent, but easier to use, definition of MIMO latency is developed in Appendix A.7. Appendix A.10 presents derivations of expressions for MIMO latency calculation based on cell-level data. Appendix A.11 discusses the user perceived delay in data communication networks. Appendix A.12 references other delay metrics.

A.2. Usual Frame Latencies as Metrics for ATM Switch Delay

An ATM switch has to deal with both contiguous and discontinuous frames. This is because ATM switches do cell-switching, *i.e.,* an ATM switch may transmit a received cell of any frame without first waiting for other cells of that frame to arrive. Thus, frames sent and received in an ATM environment are not always contiguous. Even if the input frame is contiguous, the ATM switch may transmit discontinuous frames, *i.e.,* it may introduce idle periods, unassigned cells and/or cells of other frames between cells of the frame.

The above factors make the usual frame latency metrics inappropriate for ATM switches. In this section, we show why LIFO, FIFO and FILO latencies are not appropriate metrics for an equipment perspective. Later in this appendix, we shall show that FILO latency is an appropriate metric for user perceived performance.

A.2.1. LIFO Latency

In [11], the delay in a packet-switching network is defined as the time between a "packet entry event" and a "packet exit event." A packet entry event is defined to occur at the time when the last bit of the frame enters a network, while a packet exit event is defined to occur when the first bit of the frame exits a network. This is equivalent to LIFO latency, which is considered as an appropriate metric for store-and-forward packet-switching networks because:

- packets (frames) are contiguous on both input and output and
- it is accepted that the transmission delay during packet input is an intrinsic delay for a store and forward device, for which the switch should not to be penalized.

Newer networking devices are not necessarily store-and-forward. Some of them are cut-through devices that start emitting the frame before it is received completely. Figure A.3 illustrates the case of a frame passing through a cut-through switching device with three of the four usual latencies indicated. LIFO latency is not shown because the first bit of the frame exits before the last bit of the frame enters and the LIFO latency is negative. This is a common case with cut-through devices. Thus, LIFO latency is not a good indicator of the switch delay for any cut-through type device, and as such it is inappropriate for an ATM environment, where cut-through forwarding of frames is the normal mode of operation.

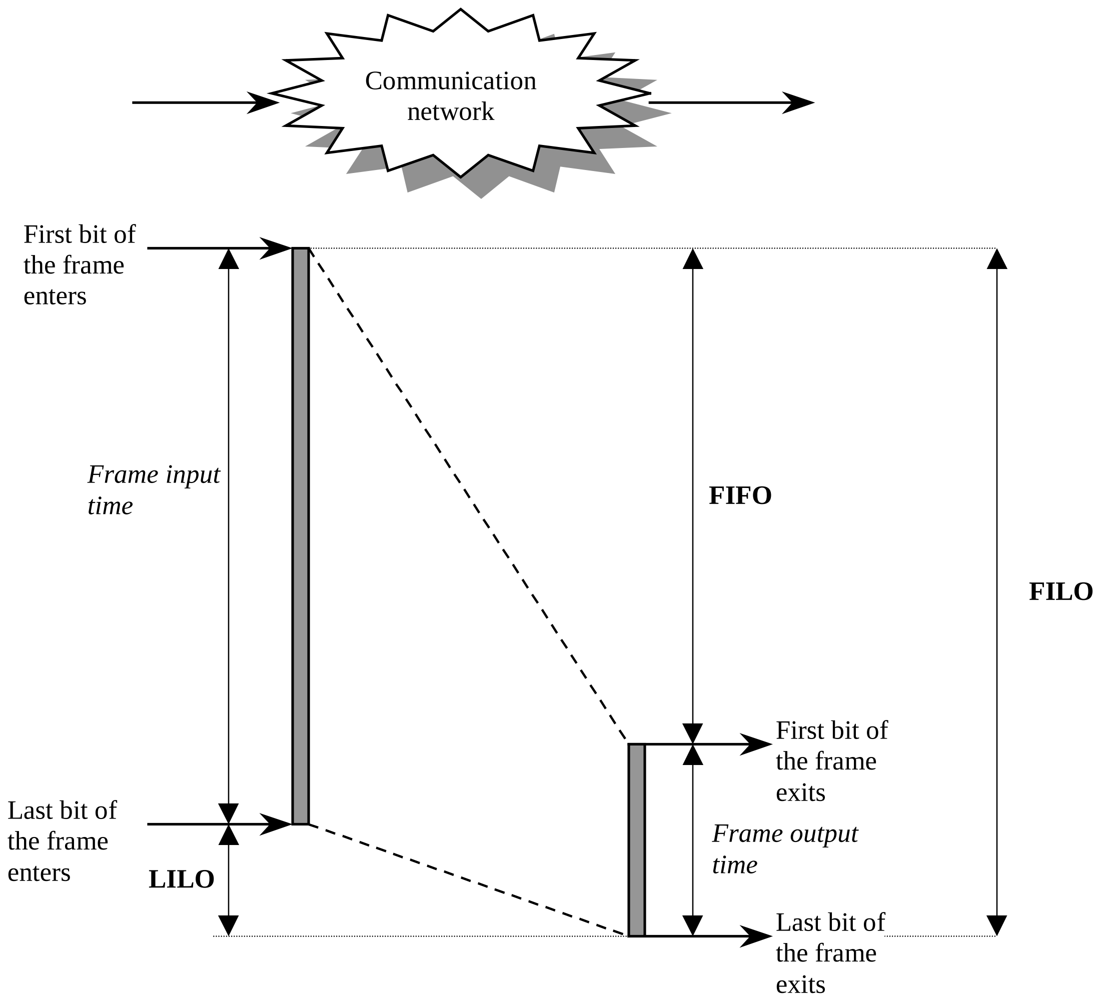

Communication network
First bit of the frame enters
Frame input time
Last bit of the frame enters
LILO
FIFO
FILO
First bit of the frame exits
Frame output time
Last bit of the frame exits

Figure A.3: Latencies of a Frame passing through a Cut-Through Switching Device

A.2.2. FIFO Latency

It is interesting to note that [17] provides a LIFO latency definition as the delay metric for store and forward switching devices, as well as a FIFO latency definition for bit forwarding devices (*i.e.*, cut-through switching devices). The introduction of FIFO latency as a delay metric is an attempt to avoid negative values for the delay through cut-through devices.

While FIFO latency may provide meaningful results if the frames are continuous, it may provide useless results if the frames are discontinuous. It is possible to have a very low FIFO delay while delays for the other parts of the frames are high. Again, since frames on ATM networks are generally discontinuous, FIFO latency is not a meaningful measure of frame latency. Figure A.4 illustrates this point.

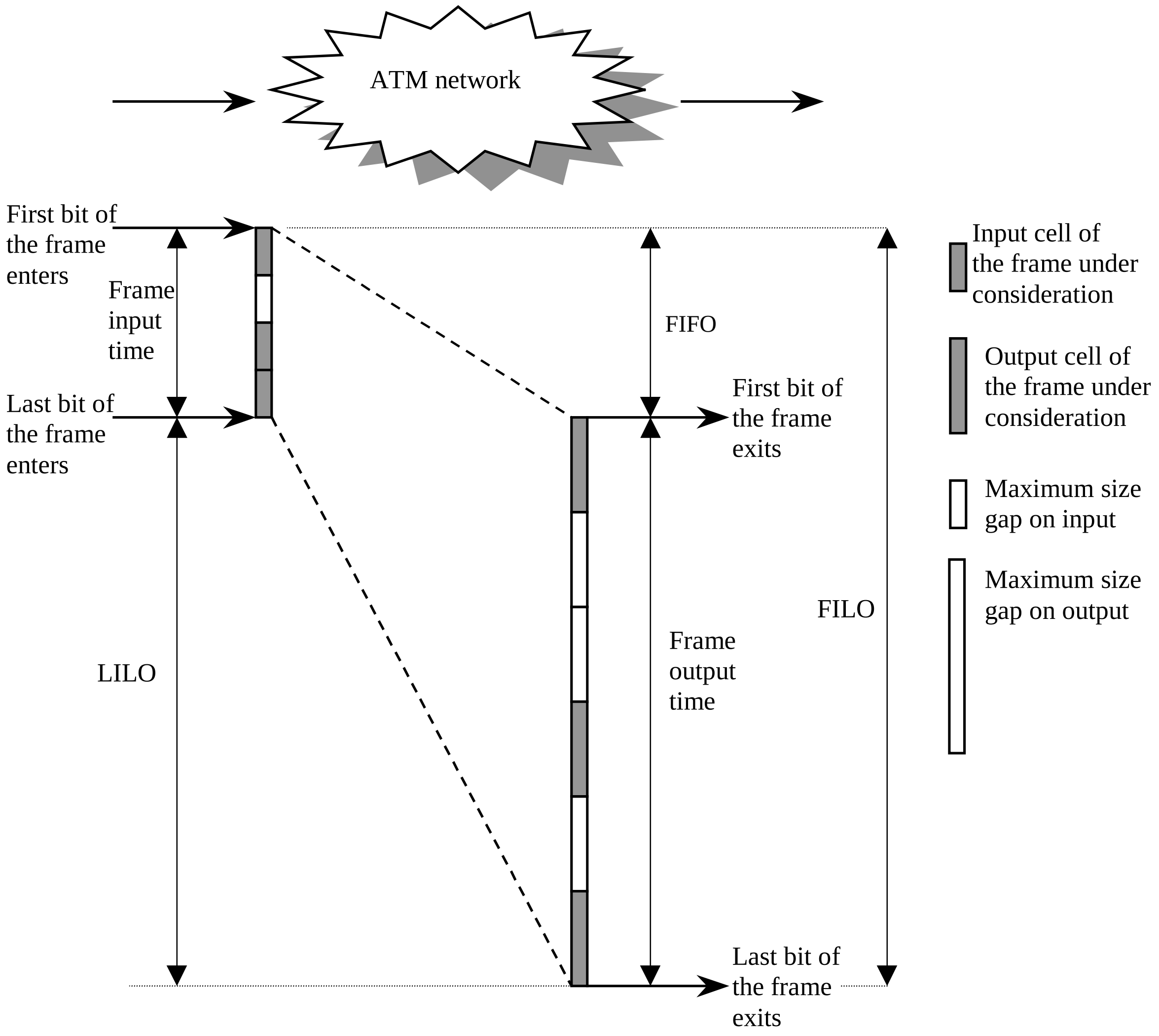

ATM network
First bit of the frame enters
Frame input time
Last bit of the frame enters
LILO
FIFO
First bit of the frame exits
Frame output time
FILO
Last bit of the frame exits
Input cell of the frame under consideration
Output cell of the frame under consideration
Maximum size gap on input
Maximum size gap on output

Figure A.4: Usual Latencies in an ATM Environment

In this case, the frame consists of 3 cells passing through an ATM switch with the input link rate higher than the output link rate. The frame is discontinuous on both input and output. The last cell is delayed considerably more than what FIFO latency would indicate.

It is possible to have one pattern of idle periods or unassigned cells (positions and a number of them) on the input of a given frame, and a completely different pattern on the output of the same frame. Note that it is also possible for a switch to remove idle periods or unassigned cells from the input, "transmitting" fewer of them on output, as we shall illustrate later.

In Figure A.4, as well as in the rest of this appendix, an unassigned cell, an idle period or a cell of another frame between cells of a given frame is indicated as a gap. In Figure A.4 the frame on input has a one-cell gap after the first cell of the frame, followed by the two remaining cells of the frame. On output, there is a two-cell gap after the first cell and then a one-cell gap between the second and the third cell of the frame.

From Figure A.4, it can be observed that it is possible for a switch to have a small FIFO latency if the first cell of a frame is transmitted quickly. However, if the later cells are delayed considerably, the receiver is not able to assemble the frame. FIFO latency does not reflect the expansion and compression of gaps on output. This is why FIFO latency is not an appropriate delay metric for switches in the ATM environment.

A.2.3. FILO Latency

From any of the previous three figures it can be noted that the relationship between FILO and LILO latency is as follows:

$$\text{FILO latency} = \text{LILO latency} + \text{Frame Input Time}$$

FILO latency is different for different frame input patterns. Suitability of LILO and FILO metrics under various circumstances is discussed after introducing MIMO latency in the next section.

A.3. MIMO Latency Definition

MIMO latency (Message-In Message-Out) is a performance metric that defines the delay introduced upon a frame passing through a switch (or any other network component). When applied to a single switch, the MIMO latency accounts only for delays introduced by the switch (because of switching and other processing) and is independent of the frame input time, output transmission time, and other physical layer delays introduced on the input and output links.

Succinctly, MIMO latency is defined as follows:

$$\text{MIMO latency} = \text{FILO latency} - \text{FILO}_0$$

where

- FILO_0 is equal to the FILO latency of a given frame passing through *a zero-delay switch*.

We define a zero-delay switch as a switch that handles incoming frames in such way that they are transmitted on the output link without any time consuming processing.

The above definition implies that MIMO latency is the difference between the measured FILO latency of a frame passing through the given switch and the FILO latency of the same frame passing through a zero-delay switch. As defined, MIMO latency has the desired property of always being positive (or zero for a zero-delay switch).

The MIMO latency is not limited to switches. It applies to all types of communication devices, including repeaters, multiplexers, (store-and-forward or cut-through) bridges, routers, ATM switches, wires, or any combination of these. MIMO latency also accounts for discontinuous frames on the input and/or output. For discontinuous frames on input, gaps may include idle periods, unassigned cells and/or cells from other frames. For discontinuous frames on output, it is assumed that there are no cells from other frames inserted between the cells of the given frame, but idle periods or unassigned cells are allowed. It should be realized that the last assumption does not present a limitation for measurements in benchmarking environments.

In the following two sections, we explore the concept of a zero-delay switch in depth.

A.4. Cell and Contiguous Frame Latency Through a Zero-Delay Switch

Figure A.5 illustrates the latency that one-bit frame would experience while passing through a zero-delay switch. As expected, a zero-delay switch should start transmission on the output link as soon as the bit arrives on the input link. Thus, the latency of a single bit through a zero-delay switch is equal to zero. A wire of a zero length is one example of a zero-delay switch.

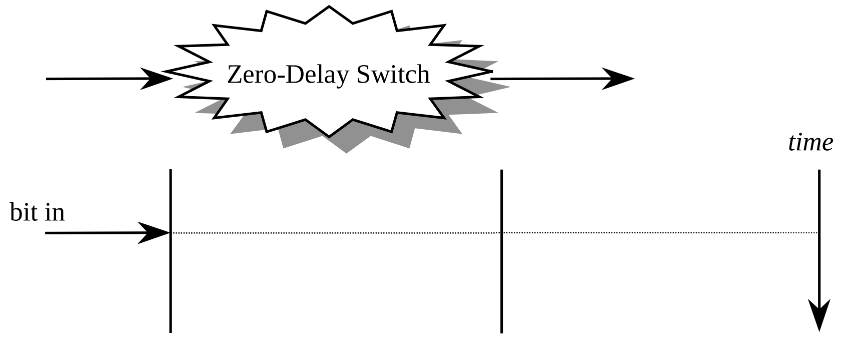


Figure A.5: Latency of One Bit passing Through the Zero-Delay Switch

Figure A.6 illustrates how a zero-delay switch would handle a cell consisting of multiple bits. The desired performance depends upon the relationship between the input and output link rates. In the case when the input link rate is equal to the output link rate, as presented in Figure A.6a, a zero-delay switch transmits each bit as soon as it arrives. Thus, each bit of the cell experiences zero latency in a zero-delay switch.

Figure A.6b illustrates the case when the input link rate is higher than the output link rate. In this case, outputting (transmitting) a bit takes longer than inputting it. The zero-delay switch can transmit only the first bit as soon as it is received. The other bits of the cell can not be transmitted immediately as they arrive, because the transmission of all previously received bits has not yet finished. Bits at the end of the cell wait longer then bits at the beginning. Thus, a zero-delay switch in this situation should be intelligent enough to do appropriate buffering of incoming bits. A zero-length wire with a FIFO buffer is an example of a zero-delay device that can handle inputs faster than the output.

Figure A.6c illustrates the case when the input link rate is lower than the output link rate. A zero-delay switch does not start transmission of the first bit immediately after it is received, but after an appropriate delay. Bits at the beginning of the cell are delayed more than bits at the end, with larger delays for slower output link rates. Only the last bit of a cell has no delay and it is transmitted immediately upon its arrival. Thus, a zero-delay switch would be intelligent enough to avoid under-runs by appropriately delaying the transmission of incoming bits. A zero-length wire with an “intelligent” FIFO buffer is an example of such a zero-delay device.

It should be realized that the illustrations in Figure A.6 apply not only to cells, but also to contiguous frames passing through a zero-delay switch.

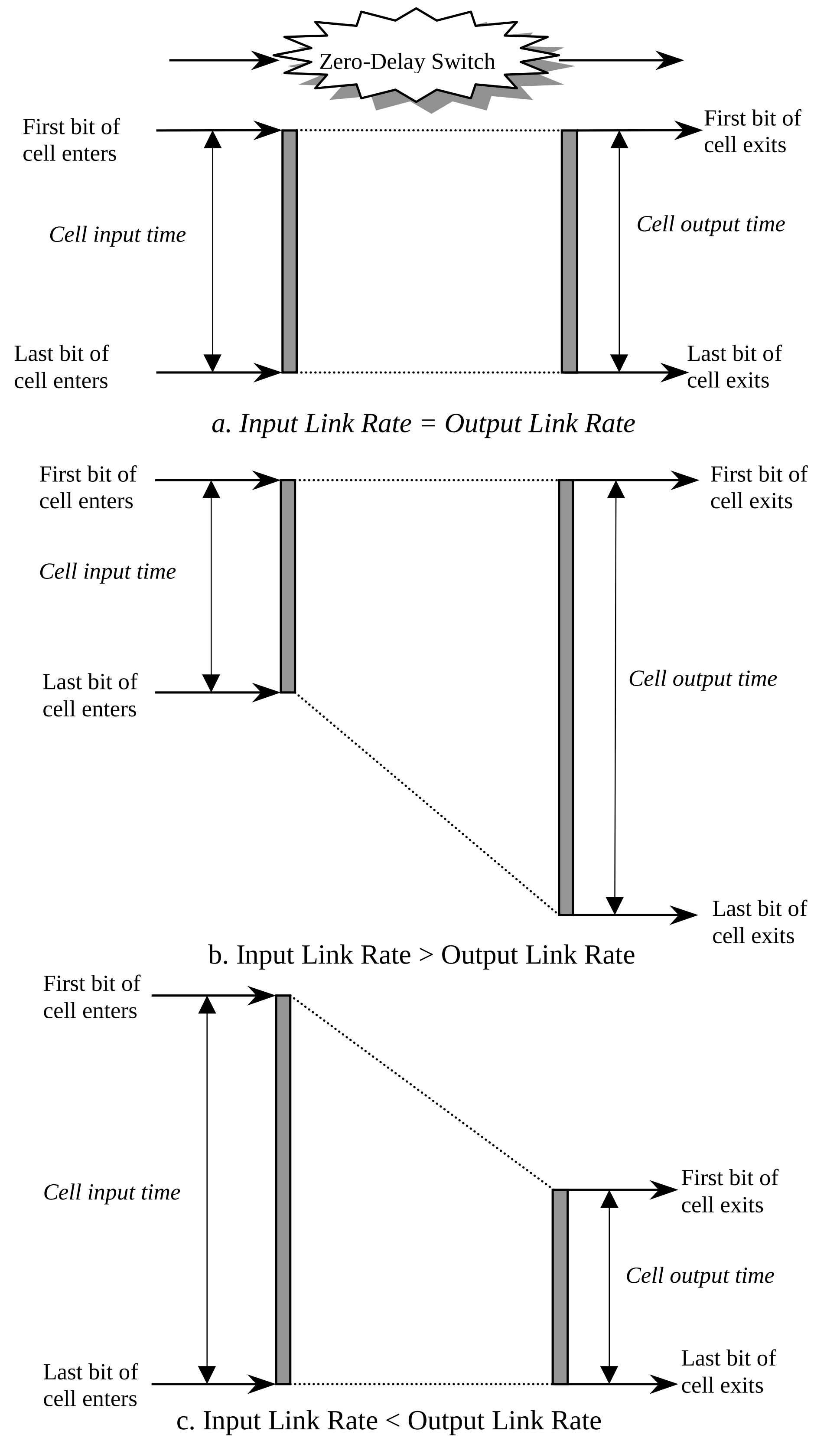
Note that a repeater can be considered as a zero-delay switch with input link rate equal to output link rate. Thus, Figure A.6a illustrates how a repeater handles incoming frames.

Also, note that a multiplexer, with n links on input and the output link capacity equal to the sum of input link capacities, can be considered as a zero-delay switch with input link rate lower than output link rate. For a multiplexer with two input links of rates equal to one half of the output link rate, Figure A.6c illustrates how the multiplexer would handle incoming frames. Similarly, a demultiplexer can be considered as a zero-delay switch with an input-link rate higher than the output-link rate. Figure A.6b illustrates operation of a two-output demultiplexer.

Based on Figure A.6, Table A.1 provides (qualitative) indications for the four usual frame latency metrics applied to a zero-delay switch. None of the latencies has a zero value in all three cases, as it should be for the latency of a frame passing through a zero-delay switch.

Table A.1: Usual Latencies Applied to a Zero-Delay Switch

	FIFO	LILO	LIFO	FILO
Input rate = Output rate	0	0	negative	positive
Input rate > Output rate	0	positive	negative	positive
Input rate < Output rate	positive	0	negative	positive



t

Figure A.6*:* Latency of One Cell passing Through a Zero-Delay switch

A.5. Latency of Discontinuous Frames Passing Through a Zero-Delay Switch

In this section, we consider how a zero-delay switch handles discontinuous frames in an ATM environment. In particular, we are interested in FILO latency, since it is used in the MIMO latency definition.

Figure A.7 illustrates one of two possible cases of a frame passing through a zero-delay switch with an input link rate higher than the output link rate. The frame includes two cells and the input link rate is 4 times the output link rate. The two cells start arriving at time t = 0 and t = 5, respectively. A zero-delay switch will start transmitting the first cell at time t = 0 and finish at time t = 4. The second cell can be transmitted without waiting and it is finished at t = 9. This is how long a zero-delay switch will take to transmit this frame. Hence, FILO latency of a zero-delay switch for this frame is 9. This is the normalized frame output time ($FILO_0$) for this input pattern. No device can transmit this frame any faster. If a device takes longer, the difference between the FILO latency of the device and $FILO_0$ is considered as the delay introduced by the device.

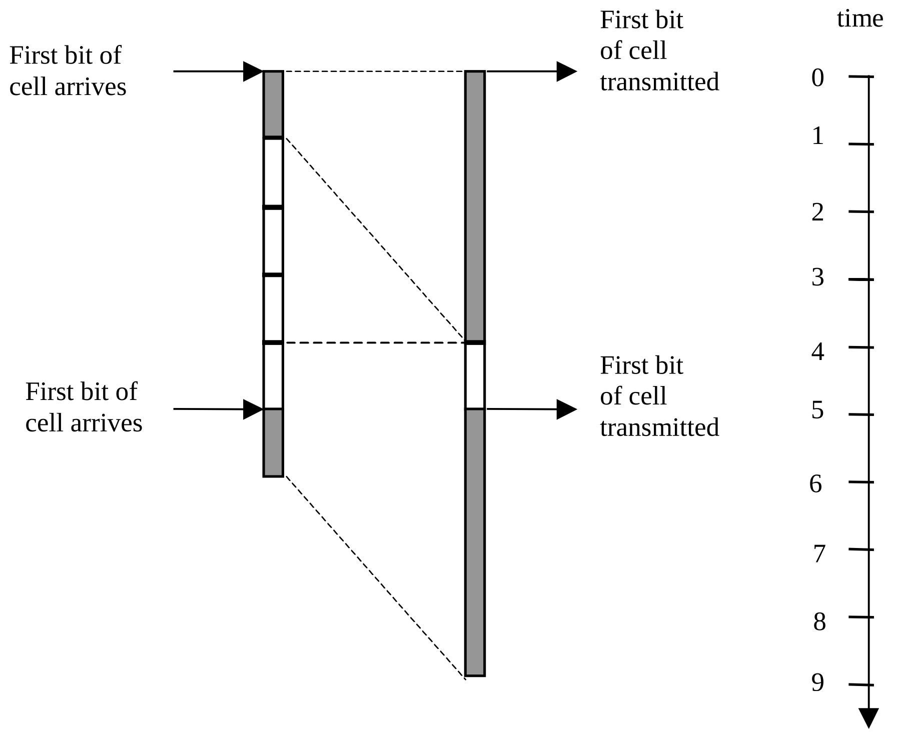

First bit of cell arrives
First bit of cell transmitted
time
0
1
2
3
4
5
6
7
8
9
First bit of cell arrives
First bit of cell transmitted

Figure A.7: Zero-Delay Switch Operations, no Cell Waiting Case (Input rate > Output Rate)

Figure A.8 shows the other possible case of a frame passing through a zero-delay switch with an input link rate higher than the output link rate. As in Figure A.7, the frame has two cells and the input link rate is 4 times the output link rate. However, the frame has a different gap pattern. The second cell arrives at time t = 2 and thus has to wait. A zero-delay switch will start transmitting the first cell at time t = 0 and finish at time t = 4. The second cell can be transmitted at t = 4 and finished at t = 8. Hence, FILO latency of a zero-delay switch for this frame is 8.

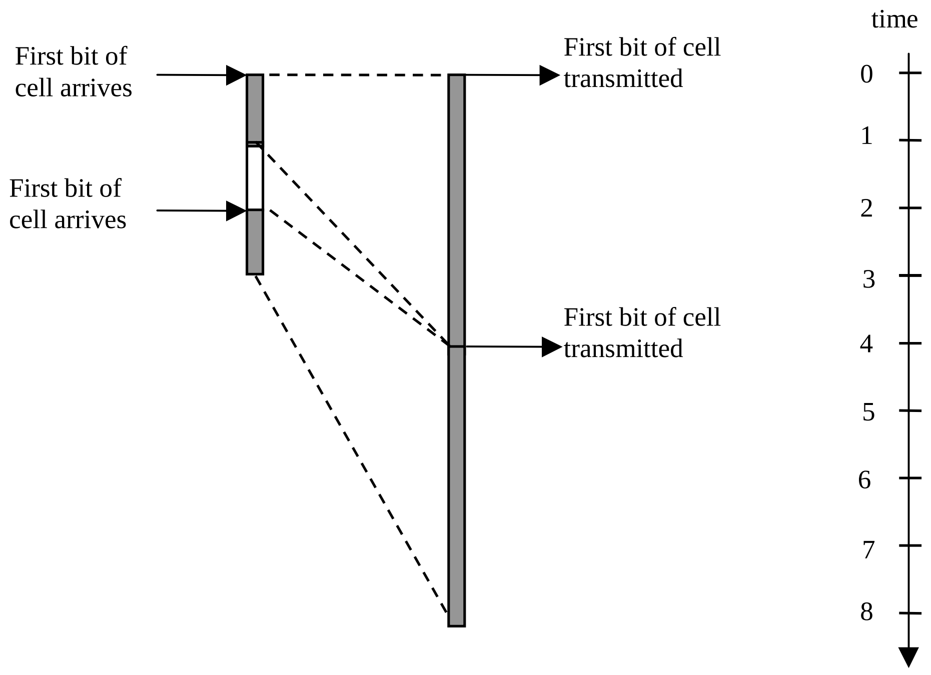

time
First bit of cell arrives
First bit of cell arrives
First bit of cell transmitted
First bit of cell transmitted
0
1
2
3
4
5
6
7
8

Figure A.8: Zero-Delay Switch Operation, Cell Waiting Case
(Input Rate > Output Rate)

Thus, in the case when the input link rate is higher than the output link rate, it is possible that:

- an incoming cell can be transmitted immediately (no cell waiting case), or
- an incoming cell has to wait for previously received cells of the same frame to be transmitted (cell waiting case).

Thus, for a given discontinuous frame, it is possible that some cells have to wait on previously received cells of the same frame, while some cells can be transmitted without waiting. Also, notice that a zero-delay switch is decreasing the size of each gap from the input, with some gaps being completely removed.

Figure A.9 illustrates the only possible case of a frame passing through a zero-delay switch with an input rate lower than the output rate. Again, the frame includes two cells but the output link rate is now four times the input link rate. The two cells arrive at time t = 0 and t = 5, respectively. A zero-delay switch will start transmitting the first cell at time t = 3 (not at t = 0, in order to avoid an underrun), and finish at time t = 4. The second cell starts at t = 8 and finishes at t = 9. This is how long a zero-delay switch will take to transmit this frame. Hence, the FILO latency of a zero-delay switch for this frame is 9.

Note that in the case when the input rate is lower than the output rate, a cell never has to wait for completion of transmissions of previously received cells. Also, notice that in this case, a zero-delay switch does not eliminate any gaps from the input, although each gap is enlarged on output. Additionally, when back-to-back cells are received on the input, new gaps are introduced between cells on the output.

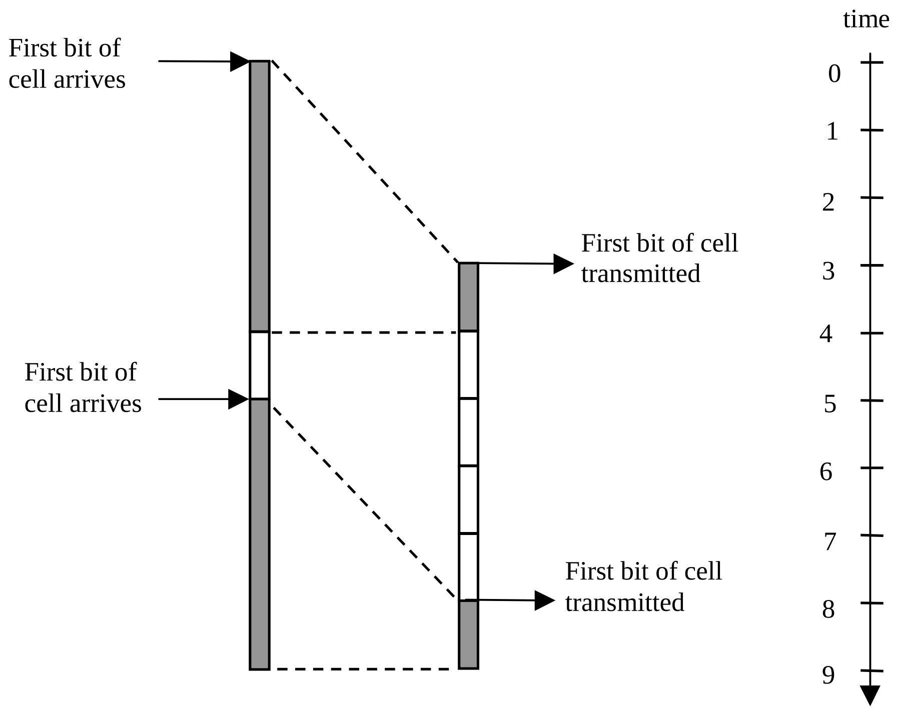

time
First bit of cell arrives
First bit of cell arrives
First bit of cell transmitted
First bit of cell transmitted
0
1
2
3
4
5
6
7
8
9

Figure A.9: Zero-Delay Switch Operation
(Input Rate < Output Rate)

A.6. Calculation of FILO Latency for a Zero-Delay Switch

The MIMO definition introduces $FILO_0$ as the FILO latency of a frame passing through a zero-delay switch. In this section, we explain how to obtain $FILO_0$ "on the fly," *i.e.*, when a frame pattern is not known in advance, but cell arrival times can be obtained in real time. We define the following parameters:

- CIT = cell input time = 424 [bits] / Input Link Rate [bits/sec]
- COT = cell output time = 424 [bits] / Output Link Rate [bits/sec]

The procedure for $FILO_0$ calculation is as follows:

a. Initially $FILO_0 = 0$ and time t is measured from the arrival of the first bit of the first cell in a zero-delay switch.

b. For each cell with its first bit arriving at time t, update $FILO_0$ as follows:

$$FILO_0 = \max\{t, FILO_0\} + CT$$

where:

$$CT = \begin{cases} CIT & \text{if input link rate} \le \text{output link rate} \\ COT & \text{if input link rate} \ge \text{output link rate} \end{cases}$$

A.7. Equivalent MIMO Latency Definition

An equivalent MIMO latency definition, which is more convenient for use in frame latency measurements and calculations when the input link rate is lower than or equal to the output link rate, can be derived as follows.

Input link rate ≤ output link rate, implies that CIT ≥ COT. A zero-delay switch will transmit the last bit of each cell of the frame as soon as it is received. In particular, the last bit of the frame is transmitted as soon as it is received. Thus, FILO_0 in these cases is equal to the frame input time:

$$\text{FILO}_0 = \text{Frame Input Time}$$

and,

$$\begin{aligned}\text{MIMO latency} &= \text{FILO latency} - \text{FILO}_0\\ &= \text{FILO latency} - \text{Frame Input Time}\\ &= \text{LILO latency}\end{aligned}$$

Then the equivalent MIMO latency definition is:

$$\text{MIMO latency} = \begin{cases}\text{LILO latency} & \text{if Input Link Rate} \le \text{Output Link Rate}\\ \text{FILO latency} - \text{FILO}_0 & \text{otherwise}\end{cases}$$

Throughout this discussion, we assume that the link rates are used in latency computation. If other rates are used, there is the potential for strange results. For example, it is possible that a carrier may offer a lower rate contract to a customer on a higher rate link. If the peak cell rate for the traffic contract is less than the link rate, and this peak cell rate is used for MIMO calculations, then the MIMO value may be negative, depending on the scheduling of cells on the link and the traffic contract. Using the link rate in MIMO calculations avoids this potential problem.

A.8. An Alternative Definition of MIMO

MIMO latency of a switch is defined as:

$$\text{MIMO} = \text{FILO} - \text{FILO}_0$$

Where FILO is the measured first-bit-in to last-bit-out latency and FILO_0 is the FILO latency of an ideal switch for the same input pattern. Note that FILO is the sum of frame input time (first bit in to last bit in) and the LILO (last bit in to last bit out) latency (see Figure A.10):

$$\text{FILO} = \text{Frame Input Time} + \text{LILO}$$

and

$$\text{FILO}_0 = \text{Frame Input Time} + \text{LILO}_0$$

Since the frame input time does not depend upon the switch, MIMO can also be expressed as:

$$\text{MIMO} = \text{LILO} - \text{LILO}_0$$

Here, LILO is the measured LILO latency and $LILO_0$ is the LILO latency of an ideal switch for the same input pattern. Given input and output speeds, $LILO_0$ can be easily computed. Figure A.11 shows three possible cases. The figure shows that $LILO_0$ is zero unless input speed is faster than the output speed.

$$LILO_0 = 0 \quad \text{if input speed} \leq \text{output speed}$$

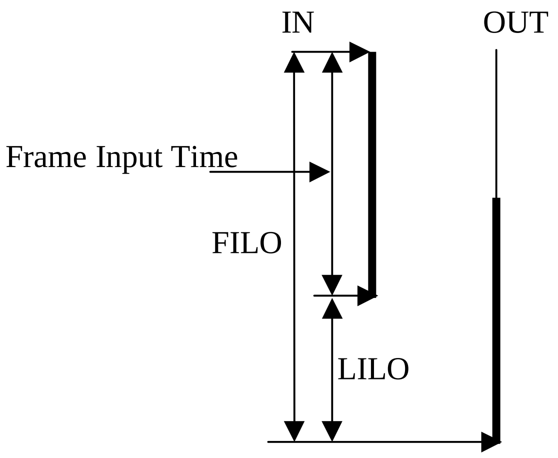
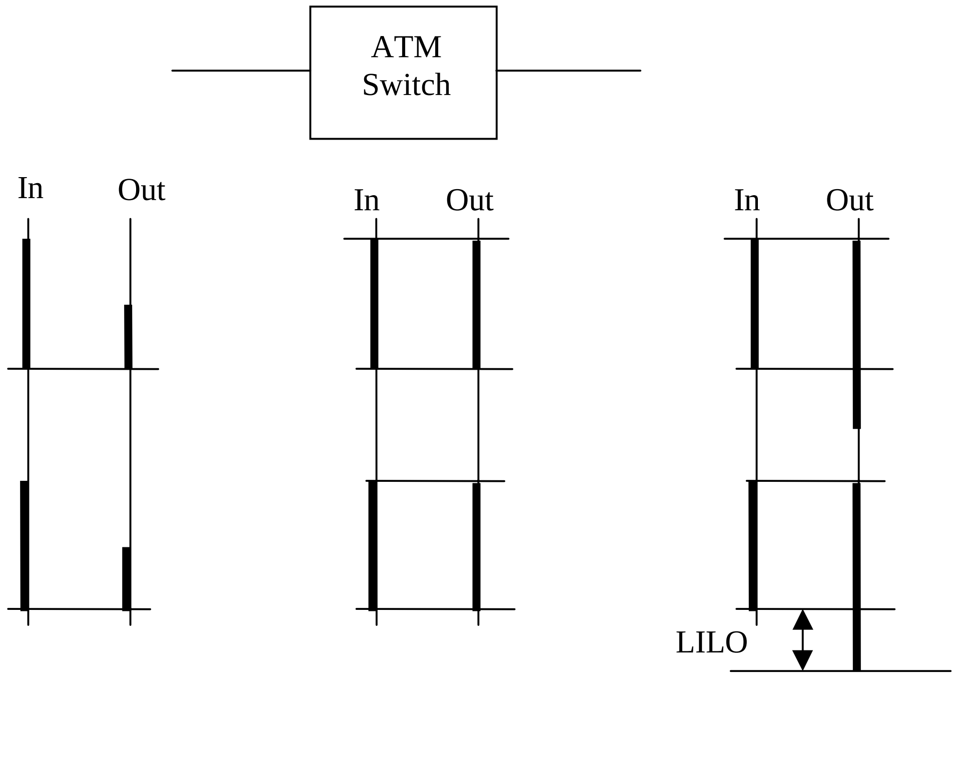


Figure A.10: Relation between MIMO and LILO



a. Input Speed < Output Speed **b**. Input Speed = Output Speed **c**. Input Speed > Output Speed

Figure A.11: An ideal switch introduces a nonzero LILO latency only when input link speed is greater than the output link speed.

A.9. MIMO Latency of a Path

Consider a network path consisting of n components in a series. Subscript i is used for the latency of the ith component and subscript Σ for the combination. Thus,

$$\text{MIMO}_i = \text{LILO}_i - \text{LILO}_{0i}$$

Similarly for a network path:

$$\text{MIMO}_\Sigma = \text{LILO}_\Sigma - \text{LILO}_{0\Sigma}$$

Since LILO is additive:

$$\text{LILO}_\Sigma = \Sigma\ \text{LILO}_i$$

The above three relationships lead us to the following identity:

$$\text{MIMO}_\Sigma + \text{LILO}_{0\Sigma} = \Sigma\ (\text{MIMO}_i + \text{LILO}_{0i})$$

Or

$$\text{MIMO}_\Sigma = \Sigma\ \text{MIMO}_i + \Sigma\ \text{LILO}_{0i} - \text{LILO}_{0\Sigma}$$

This relationship allows us to compute MIMO of a series of components from the measured MIMO values of individual components. Note that LILO_{0i} and $\text{LILO}_{0\Sigma}$ can be computed given the input pattern and the input output speeds.

We illustrate this with a few examples.

Example 1:

Consider the configuration shown in Figure A.12 consisting of two switches interconnected via a wire. All links and ports are 150 Mbps. The input frame is composed of two cells with a gap of 3 cell times. Let us suppose that each switch introduces a MIMO equal to c, where c is the cell time at 150 Mbps. Also, for simplicity assume that the wire between the switches also introduces a MIMO of c. (Other wire lengths can be handled similarly).

Since all input/output speeds are the same, an ideal switch will produce zero LILO latency. Hence,

$$\text{LILO}_{0i} = 0$$
$$\text{LILO}_{0\Sigma} = 0$$
$$\text{MIMO}_\Sigma = \Sigma\ \text{MIMO}_i = c + c + c = 3c$$

That is, the MIMO latency is simply the sum of the individual MIMO latencies.

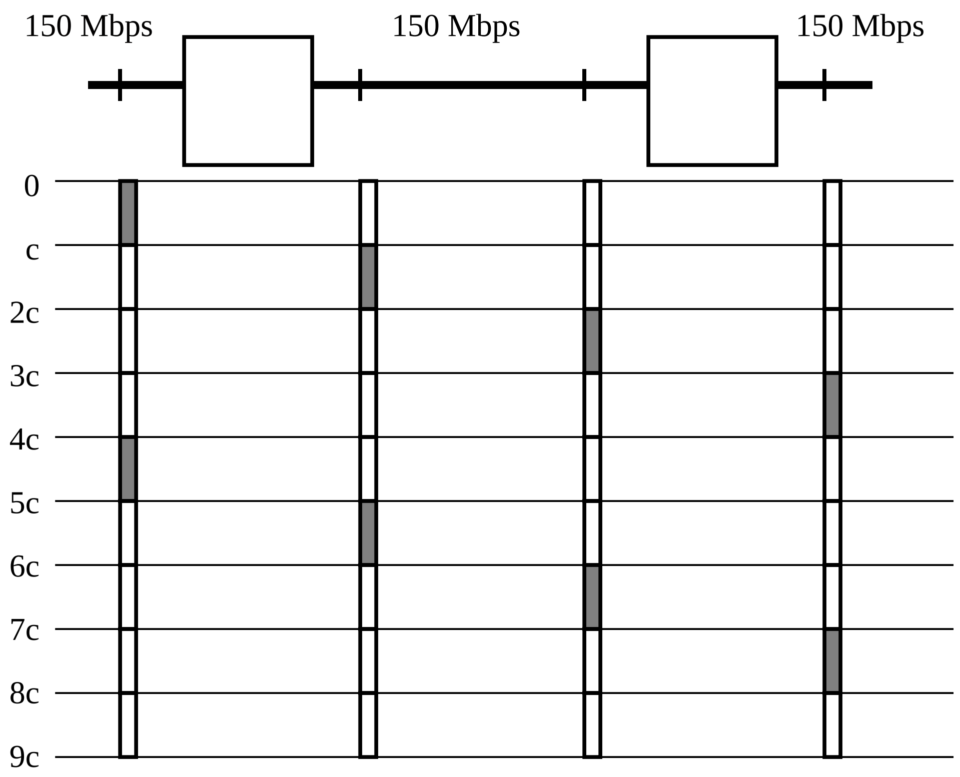


Figure A.12: MIMO Aggregation for Example 1.

Example 2:

Consider the configuration shown in Figure A.13. This is similar to the configuration of Example 1, except that the intermediate link is 50 Mbps and therefore, introduces a delay of 3c.

In this case, the first switch has an input speed of 150 Mbps, while the output speed is 50 Mbps. An ideal switch with these I/O speeds will produce a LILO latency of 2c, where c is the cell time at 150 Mbps. That is,

$$\text{LILO}_{01} = 2c$$

For the wire as well as the second switch, the input speed is equal to or less than the output speed and so the LILO_0 is zero:

$$\text{LILO}_{02} = 0$$
$$\text{LILO}_{03} = 0$$

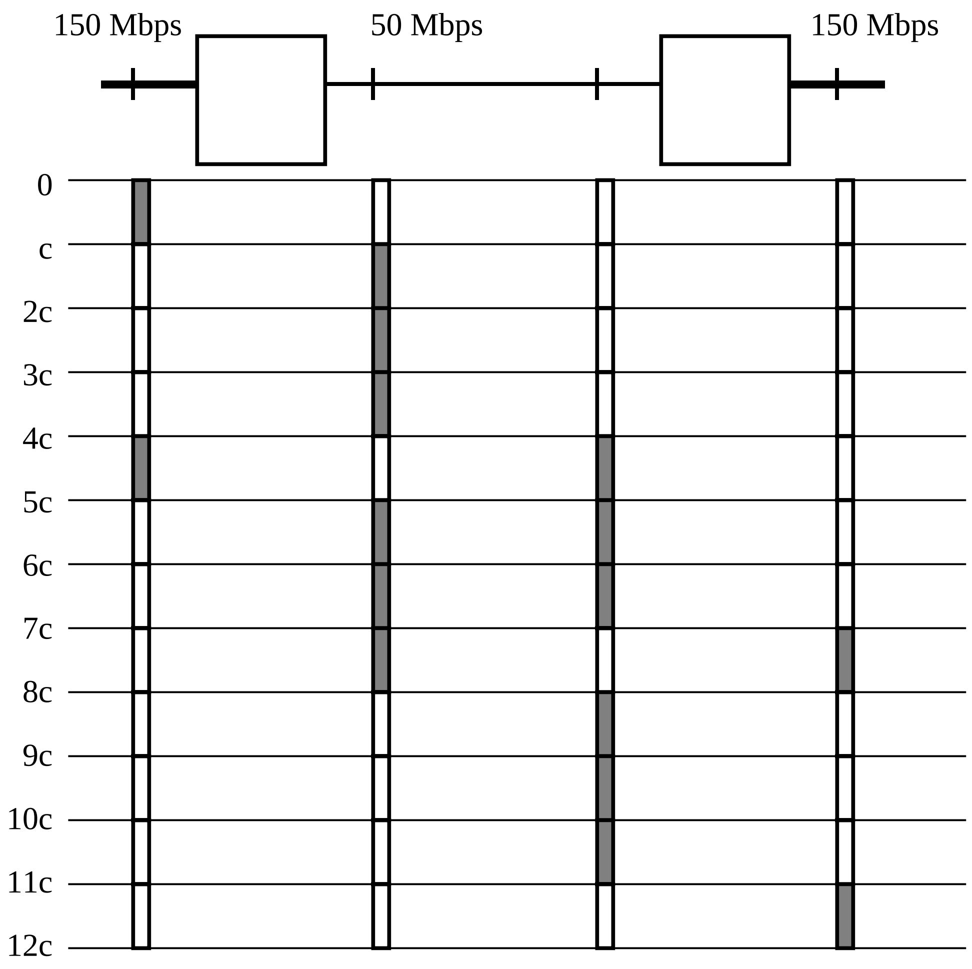


Figure A.13: MIMO Aggregation for Example 2.

If the whole network is replaced by a single ideal switch, that switch will have an input speed of 150 Mbps and an output speed of 150 Mbps and therefore, will have a zero LILO latency.

That is,

$$LILO_{0\Sigma} = 0$$

Using the above values, we get:

$$MIMO_{\Sigma} = \Sigma\ MIMO_i + \Sigma\ LILO_{0i} - LILO_{0\Sigma} = (c+3c+c) + (2c+0+0) - 0 = 7c$$

A.10. Measuring MIMO Latency

To measure MIMO latency for a frame passing through the System Under Test (SUT), the times of occurrence for the following two events need to be recorded:

- the first-bit of the frame enters into the SUT,
- the last-bit of the frame exits from the SUT.

The time between these two events is the FILO latency. $FILO_0$ can be obtained from the cell pattern of the test frame on input as explained in Appendix A.6. Substituting FILO latency and $FILO_0$ into the MIMO latency formula would give the SUT's delay for a given frame.

If the input link rate is lower than or equal to the output link rate, it is easier to calculate MIMO latency. In this case, the times of occurrence for the following two events need to be recorded:

- the last-bit of the frame enters into the SUT, and
- the last-bit of the frame exits from the SUT.

The time between these two events is the LILO latency, which is equal to the MIMO latency for the frame. Note that the cell arrival pattern does not matter in this case.

Contemporary ATM monitors provide measurement data at the cell-level. Considering that the definition of MIMO latency uses bit-level data, we now describe how to calculate MIMO latency using measurements at the cell-level. Standard definitions of two cell-level performance metrics, which are of importance for MIMO latency calculation are:

- cell transfer delay (CTD), defined as the time between the first bit of the cell entering the switch and the last bit of the cell leaving the switch, and
- cell inter-arrival time, defined as the time between arrival of the last bit of the first cell and arrival of the last bit of the second cell.

In cases where input link rate is higher than output link rate, according to the MIMO latency definition, FILO latency has to be measured. From Figure A.14, it can be observed that:

FILO latency = First cell's transfer delay + First cell to last cell inter-arrival time

Thus, to calculate MIMO latency when the input link rate is higher than or equal to the output link rate, it is necessary to measure the transfer delay of the first cell of a frame and the inter-arrival time between the first cell and the last cell of a frame.

In cases when input link rate is lower than or equal to output link rate, it is sufficient to measure LILO latency. From Figure A.15, it can be observed that:

LILO latency = Last cell's transfer delay – CIT

Thus, to calculate MIMO latency when the input link rate is lower than or equal to the output link rate, it is necessary to measure the transfer delay of the last cell of a frame.

A.11. User Perceived Delay

It should be pointed out that MIMO latency measures only the SUT's contribution to the delay. It does not include the delay caused by components not in the SUT's control. In particular, it does not include the frame input time. However, a user using the system does have to wait while the frame is being sent to the SUT. A user typically assembles the frame and gives it to the network. The user starts waiting as soon as the first bit starts entering the system and cannot do any meaningful work until the last bit exits the network. Thus, user perceived performance is reflected by FILO latency.

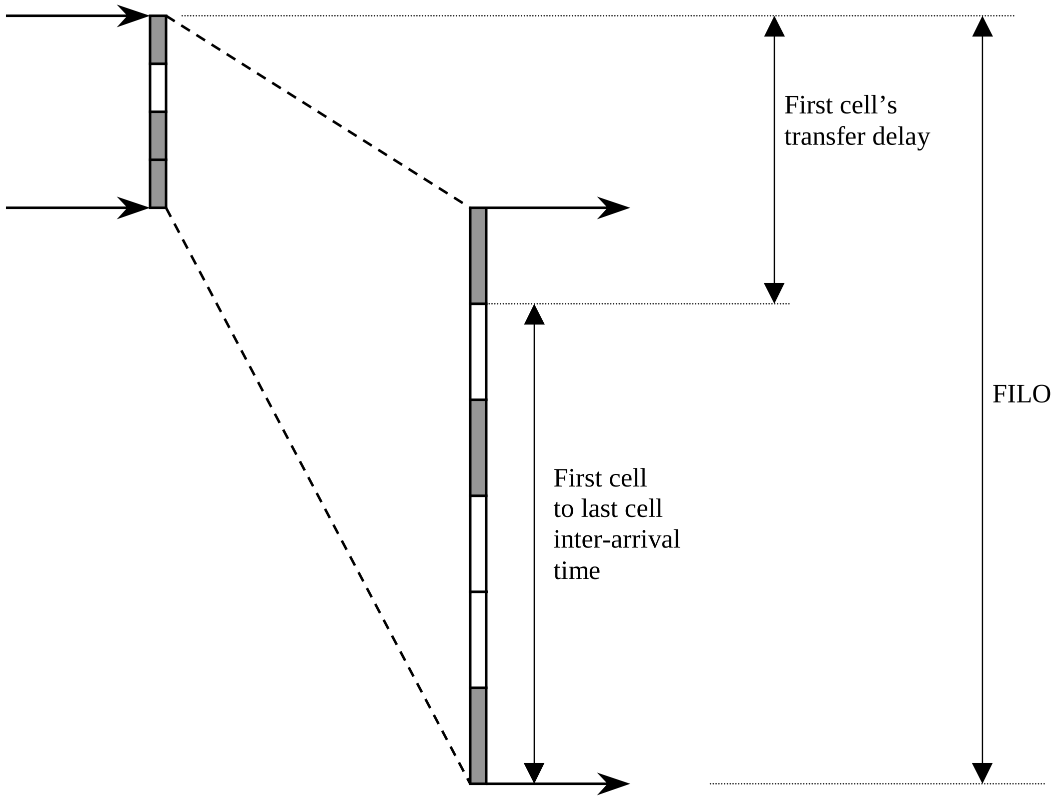


Figure A.14: FILO Latency Calculation
(Input Rate > Output Rate)

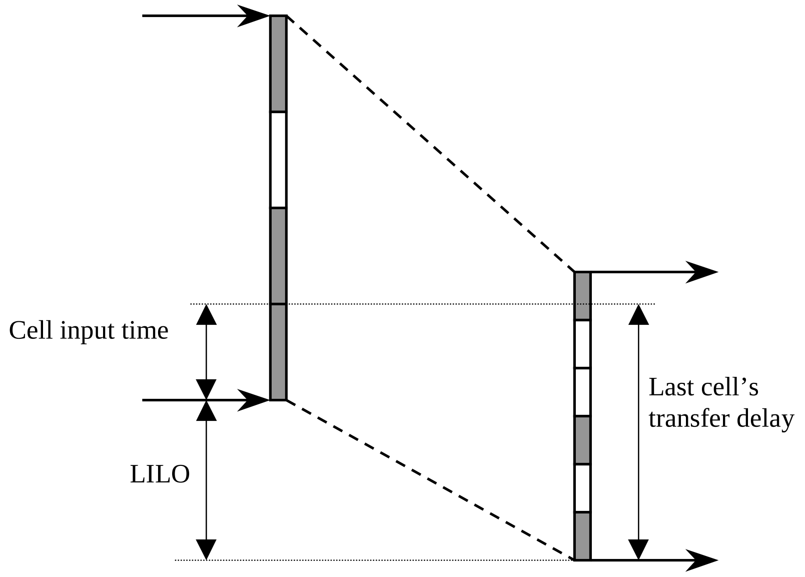


Figure A.15: LILO Latency Calculation (Input rate ≤ Output Rate)

Figure A.16 illustrates the relationships between the user perceived performance and MIMO latency in two scenarios with continuous frames. In the first scenario, the input link rate is same as the output link rate. In the second scenario, the output is slower. The switch delay, as given by MIMO latency, is same in both cases; but the user perceived delay, as given by FILO latency, is different. For the case in Figure A.16b, FILO latency is worse. It can be observed that the user perceived delay depends upon input/output link speeds. On the other hand, network delay measured by MIMO latency is independent of link speeds. The difference between those two delays is the frame latency through a zero-delay switch.

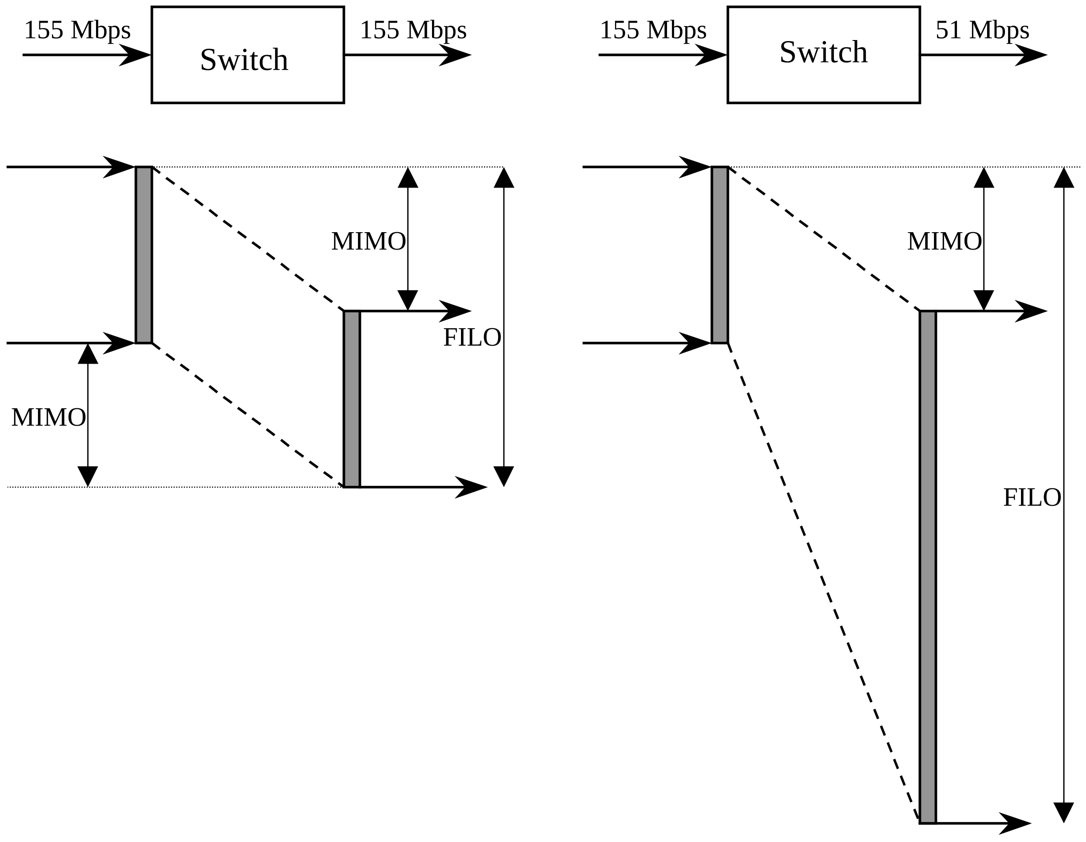


Figure A.16: FILO Latency as User Perceived Delay

A.12. Other Delay Metrics

T1A1.3 is considering a Last Cell Delay (LCD) frame latency metric that indicates the latency induced to the last cell by the SUT. LCD and LILO allow additivity and allocation of end-to-end frame latency.

Appendix B: Methodology for Implementing Scalable Test Configurations

B.1. Introduction

Throughout this appendix it is assumed, for improved readability, that the IUT consists of a single switch, although the methodologies presented here apply equally to test cases in which the IUT is a network of switches or, alternatively, a subset of modules of a single switch. The notation P_{ij} is used to refer to the j^{th} port of the i^{th} module of the IUT, and (P_{ab}, P_{cd}) indicates that a connection (either internal or external to the IUT) exists between P_{ab} and P_{cd}.

In Section 3.3, a number of connection configurations have been presented. In most of the cases, these configurations require one traffic generator and/or analyzer for each switch port. Thus, the number of generators and/or analyzers increases as the number of ports increases. It is desirable to define scalable configurations that can be used with a limited number of generators. However, one problem with scalable configurations is that there are many ways to set up the connections and measurement results could vary with the setup. For example, in the case of unicast, it may not be possible to overload a port with only one generator. Using two generators in scalable configurations may exhibit different behavior, such as overloading, that may not show up with one generator.

Performance testing requires two kinds of virtual channel connections (VCCs): foreground VCCs (traffic that is measured) and background VCCs (traffic that simply interferes with the foreground traffic). The methodology for generating configurations of both types of VCCs is covered by this appendix.

The VCCs are formed by setting up connections between ports of the switch. The connections are internal through the switch fabric and external through some transmission medium or wires (which could be cables, fibers, or even wireless links), depending on the port technology. In this Appendix, internal connections are shown by thin lines and external connections by thick lines. ATM connections (those established internal to the IUT) are inherently bi-directional. A uni-directional test configuration is one in which the flow of test traffic occurs in only one direction across each ATM connection. Alternatively, a bi-directional test configuration would exercise both directions. Whenever external connections are used in the test configurations, only permanent VCCs can be established.

Two generic categories of scalable configuration are presented in this appendix, namely:

1. "Parallel Traffic Replication" configurations, discussed in Appendix B.2, which employ the point-to-multipoint capability of a switching system (other than the IUT) to artificially generate more traffic than is possible with a limited number of traffic generators, and
2. "Serial Traffic Replication" configurations, discussed in Appendix B.3, which employ external connections to serially relay traffic egressing from the IUT back in to the IUT, thereby emulating additional traffic generators.

B.2. Parallel Traffic Replication

The point-to-multipoint capability of a switching system is a method for traffic replication intended primarily for broadcast communication services, but which also lends itself well to the task of generating the traffic inputs to the IUT that are required for the test configurations shown in Figure 3.6. Identical ATM cells are broadcast in parallel from multiple output ports – hence 'parallel traffic replication'. Given a single traffic generator, and a switching system (other than the IUT) with a point-to-multipoint capability (a multicast switch), an IUT may receive traffic on as many input ports as the multicast switch has available output ports. This form of scalable configuration is depicted in Figure B.1, where G is the single traffic generator. Internal to the IUT, any of the configurations from Figure 3.6 may be used.

Note that if the multicast switch does not support multipoint-to-point connections, then this form of parallel traffic replication cannot support bi-directional test configurations. In such cases, it may be necessary to use serial traffic replication, as described in the next section. Also, the measured results may be affected by the performance of the multicast switch.

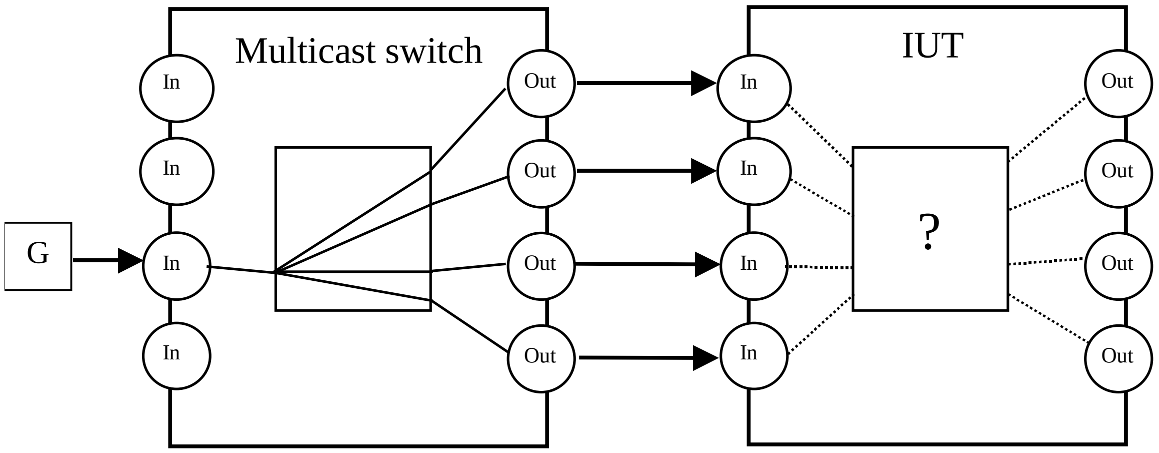


Figure B.1: A Parallel Traffic Replication Scalable Configuration

B.3. Serial Traffic Replication

The serial traffic replication method for generating large bandwidths of test traffic requires a minimal number of test ports. However serial traffic replication may lead in some cases to misleading results about the performance of the SUT. There is no guarantee that an SUT that performs poorly with serial traffic replication may perform poorly with parallel traffic replication or under real-world conditions. Interpretation of results from traffic replication configurations requires a detailed understanding of the SUT, input test sources and proper analysis techniques. For instance, the throughput of an SUT with serial traffic replication may be less than the throughput of the SUT with parallel traffic replication or real-world conditions. In such cases, the use of serial traffic replication may nullify any gains achieved by statistical multiplexing. Therefore, when using this method, caution must be taken when interpreting results for the performance of the SUT.

An example test configuration employing serial traffic replication is provided in Figure B.2, which shows a 4-port switch with ports labeled as P_{11}, P_{12}, P_{21} and P_{22}. Of these, ports P_{21} and P_{12}

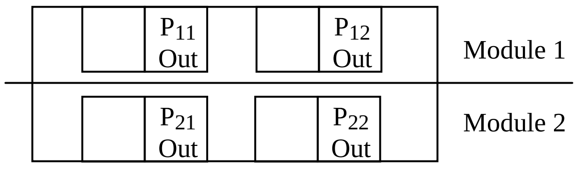
are connected by a wire W_1, while port P_{22} has a "loopback" wire LB that connects the output of the port to its input. Internally, a PVC has been set up to connect ports P_{11} with P_{21} and P_{12} with P_{22}. Note that all external connections (wires) and internal connections (PVCs) in this case are bi-directional, except the loopback. During testing with this configuration, cells first enter the switch at P_{11} and are passed through every port of the switch in series, before looping back at P_{22} and following the reverse path back to exit the switch for the last time at P_{11}.

The methodology presented here has two phases. During the first phase the switch ports are connected externally by numbered wires, as in Appendix B.3.1. The second phase consists of setting up PVCs, *i.e.*, internal connections between ports, as explained in Appendix B.3.2.

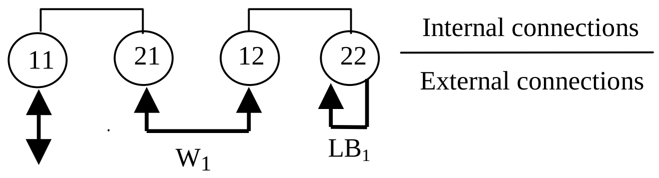
The sequence of concatenated connections (internal and external) is called a **VCC Chain**. For example, the VCC shown in Figure B.2.b is formed by setting up a VCC chain starting from P_{11}-In. ATM cells flow internally from P_{11}-In to P_{21}-Out, externally via wire W_1 to P_{12}-In, internally to P_{22}-Out, externally via wire LB to P_{22}-In, internally to P_{12}-Out, externally through wire W_1 to P_{21}-In, finally exiting at P_{11}-Out. This VCC chain can be indicated as:

generator-P_{11}-P_{21}-P_{12}-P_{22}-P_{22}-P_{12}-P_{21}-P_{11}-analyzer.

Of these connections, P_{22}-P_{22} is a unidirectional external connection (loopback, denoted as LB) and P_{12}-P_{21} is a bi-directional external connection (wire, denoted as W). The sequence of external connections used in this VCC chain is: Generator-W_1-LB-W_1-Analyzer. Both the above notations are symmetric in the sense that the second half of the chain is a mirror image of the first half. For example, W_1-LB is the mirror image of LB-W_1.



a) Representation of physical switch



b) Logical representation of the switch and its port connections.

Figure B.2: A VCC Chain that can Implement the 4–to–4 Straight Configuration.

Another possible configuration for this "n-to-n single generator scalable configuration" is P_{11}-P_{12}-P_{21}-P_{22}-P_{22}-P_{21}-P_{12}-P_{11}. The various VCC chains may be distinguished by the order of, and the direction through which, each wire is initially traversed by the generated traffic.

The four-port switch shown in Figure B.2 consists of two modules with two ports each. The measured performance for a given test configuration may depend upon whether the internal connections of the VCC chain are inter-module, intra-module, or a mixture of both. The methodology presented in this appendix ensures that it is possible for exclusively inter-module, or intra-module traffic to be carried. The trivial case occurs when the IUT consists of a single module. In such cases, all ATM connections are intra-module.

B.3.1. Implementation of External Connections

The methodology for implementing the external connections consists of the following three steps:

1. Identify the modules to be included in the IUT and label the ports (using P_{ij} format).
2. Connect the generators and analyzers to the appropriate ports.
3. Establish and number external connections (wires) to use all the remaining ports of the IUT.

These steps are now explained.

Step 1. Identifying the Modules to be Included in the IUT

In order to ensure that it is possible for the configuration to support exclusively inter-module and/or intra-module internal connections, the IUT should consist of pairs of similar modules. If this constraint is not satisfied, the VCCs that are established may be a mixture of inter-module and intra-module connections. It is not necessary that the modules/ports be labeled, although we use the P_{ij} format here to assist in the description of the methodology.

Consider a switch with several modules of different port types. The ports could be different in speed and/or connector type. Each module may have a different number of ports. For example, a switch may have two modules of eight and six 155-Mbps single-mode fiber ports, respectively, another module with eight 155-Mpbs UTP ports and a fourth module with six 25-Mbps UTP ports. Figure B.3 shows an example IUT where the modules are grouped by type. The first group consists of two 25-Mbps UTP modules, the second group consists of two 155-Mbps single fiber modules. External connections may only be established between ports that are co-located within the same group (hence the constraint that modules come in pairs for inter-module connectivity).

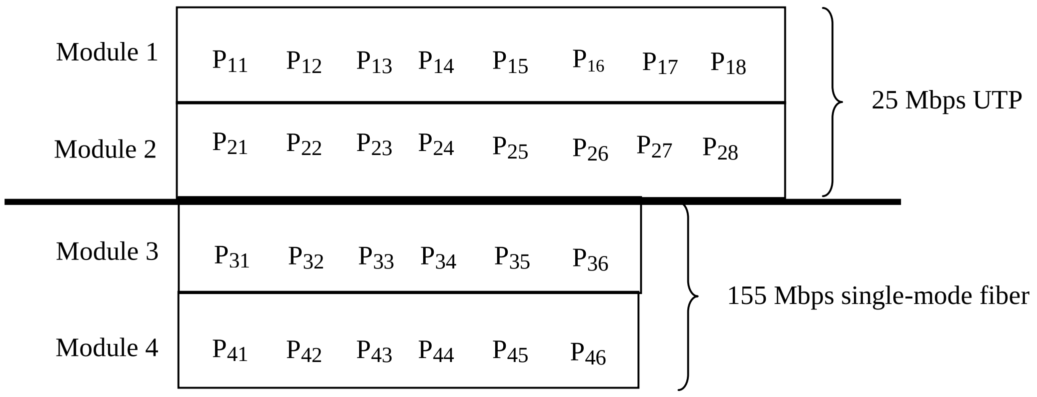


Figure B.3: Example Partitioning of Modules into Groups.

Step 2. Connect the Generators and Analyzers to the Appropriate Ports

A port must be reserved for each generator/analyzer that is to be used in the test. These reserved ports cannot be used in the next step that establishes external connections. The methodology presented here allows any given number, r<n, of generators. Some additional constraints on the number of ports in the IUT are explained in the next step.

Step 3. Establish and Number External Connections

After selecting the ports that are reserved for connection to generators/analyzers, the remaining ports have to be externally connected with numbered "wires". The following guidelines should be followed when establishing the wires:

1. Partition the remaining (non-reserved) ports into subgroups, whilst ensuring that there is an odd number of ports in each subgroup.
2. Assign each generator/analyzer to a subgroup. To establish a "straight" connection configuration it is necessary that there exist a 1-to-1 correspondence between subgroups and generators with an even partitioning of ports into subgroups. For other test configurations, such as "full cross" or "partial cross" (see Figure 3.6), more than one generator may be assigned to a subgroup, and the ports need not be evenly distributed between subgroups.
3. With m non-reserved ports in a particular subgroup (m being odd), the first m-1 ports are pair-wise connected by wires, numbered consecutively from 1 across all subgroups (*i.e.*, the 1st subgroup's wires would be W_1 to W_x, where x = (m-1)/2, and the 2nd subgroup would have wires numbered from W_{x+1}. It is preferable that wires be established between ports on different modules to ensure that exclusively inter-module or intra-module traffic may be carried. Wires cannot connect ports of different types.
4. The last (mth) port of each subgroup is occupied by a loopback, labeled as LB_g for the gth subgroup, that will redirect all traffic egressing from the port back to the ingress of the port.

The following example illustrates the methodology for establishing the wires:

Consider the n-to-n straight configuration required for the foreground traffic in throughput measurement. Suppose the switch has two modules with four ports each of the same speed and connection type. The ports are labeled as shown in Figure B.4a. P_{11} is arbitrarily selected to be connected to the single generator/analyzer that is to be used. There will only be one subgroup, because there is only one generator/analyzer and all the ports are of the same speed and connection type. The non-reserved ports in the subgroup are {P_{12}, P_{13}, P_{14}, P_{21}, P_{22}, P_{23}, P_{24}}. The wires W_1=(P_{12}, P_{21}), W_2=(P_{13}, P_{22}) and W_3=(P_{14}, P_{23}) are obtained by alternatively selecting ports of the first and second module from the non-reserved list of ports. Finally, the loopback is placed at the remaining port, namely P_{24}. Figure B.4b shows the resulting wiring configuration. Note that it will allow for either exclusively inter-module, or exclusively intra-module traffic (this will be shown in the next section).

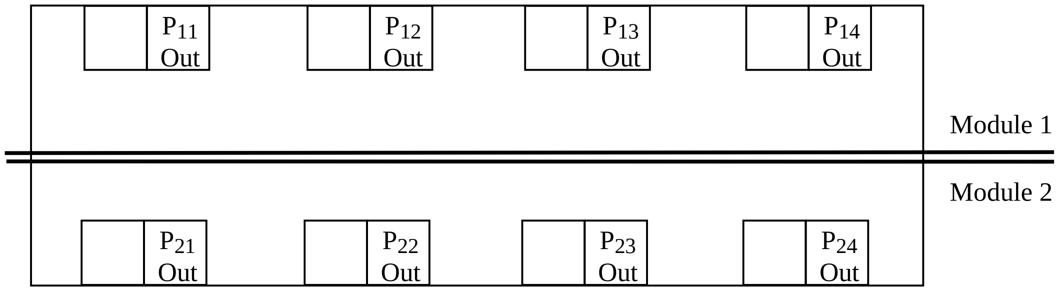


Figure B.4a: Port Labeling of the Example Switch (2 Modules of 4 Ports Each).

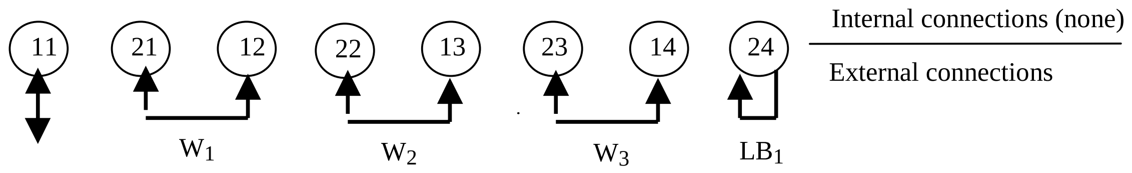


Figure B.4b: Wiring Configuration for the Example Switch.

B.3.2. Implementation of Internal Connections

The methodology presented in this appendix can ensure that exclusively inter-module or intra-modules traffic is carried, depending on the implementation of the internal connections. For example, Figures B.5a and B.5b show configurations for intra-module and inter-module configurations, respectively, although both are based on the wiring indicated in Figure B.4b.

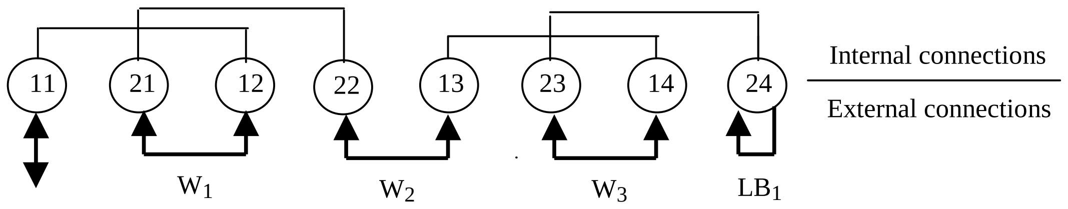


Figure B.5a: An 8-to-8 Straight Configuration for the Example Switch using Intra-Module VCCs.

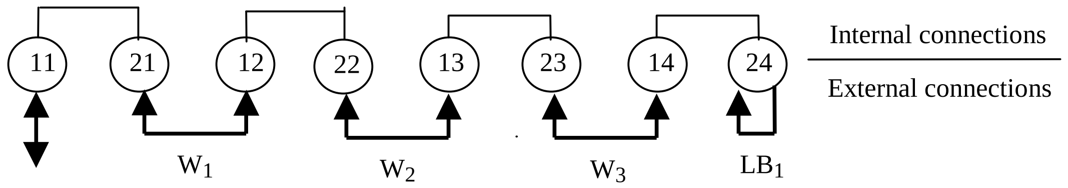


Figure B.5b: An 8-to-8 Straight Configuration for the Example Switch using Inter-Module VCCs.

As indicated earlier, VCC chains may be distinguished by the order of, and the direction through which, each wire is initially traversed by the generated traffic. We only consider VCC chains that use every non-reserved port within a single subgroup.

Using the same terminology as in Appendix B.3.1, assume that we have established x wires and a single loopback for a subgroup with m non-reserved ports, and r generators. For a given traffic flow (intra-module or inter-module), a VCC chain that passes through each non-reserved port can be expressed as:

generator – 'x wires in series' – LB – 'x wires in reverse series' – analyzer.

Stated more succinctly, the VCC chain only depends on the generator/analyzer used and the ordering of the 'x wires in series'. The example in Figure B.5 used the ordering $W_1W_2W_3$. For a given traffic generator and wire ordering there exists a unique set of VCCs that provides an intra-module VCC chain. Similarly for an inter-module VCC chain.

Without loss of generality, assume that the wire ordering is $W_1W_2 \ldots W_x$. Let P_g be the port connected to the generator for this VCC chain, and P_{LB} be the port that has the loopback. To complete the construction of the VCC chain, all that remains is to establish the VCCs, as follows:

1. Set $P_{in} = P_g$ and $i = 1$.
2. Set P_{out} = the port from W_i that is/isn't located on the same module as P_{in} for an intra/inter-module VCC chain, respectively.
3. Establish the bi-directional internal connection $V_i = (P_{in}, P_{out})$.
4. Set P_{in} to be the other port from W_i (not P_{out}).
5. If $(i < x)$, set $i = i+1$ and return to step 2, otherwise continue.
6. Establish the bi-directional internal connection $V_{x+1} = (P_{in}, P_{LB})$.

The VCC chain is now fully specified as:

$$\text{generator-}V_1\text{-}W_1\text{-}V_2\text{-}W_2\text{-} \ldots \text{-}W_x\text{-}V_{x+1}\text{-LB-}V_{x+1}\text{-}W_x\text{-} \ldots \text{-}W_2\text{-}V_2\text{-}W_1\text{-}V_1\text{-analyzer.}$$

Clearly, if any other wire ordering, or any other generator (out of the r generators assigned to the subgroup) was used, a different VCC chain would have resulted.

Referring again to the example provided in Figures B.4a and B.4b, only one VCC chain is established for each traffic flow. In the case of an n-to-m partial cross configuration with a single generator, several VCC chains are required, each based on a different ordering of the same subgroup of wires. The six (3 factorial) possible permutations for the example are: $W_1W_2W_3$, $W_2W_3W_1$, $W_3W_1W_2$, $W_3W_2W_1$, $W_2W_1W_3$ and $W_1W_3W_2$, each of which would result in a different VCC chain for the single generator.

The examples given in Appendix B.3.4 illustrate the ways in which multiple VCC chains and/or multiple generators may be used in serial traffic replication scalable configurations.

B.3.3. Background Traffic

To establish a configuration that incorporates background test traffic, the background traffic may:

1) ingress and egress on ports different from those used by the foreground traffic,
2) ingress with, but egress at a port different from, the foreground traffic, or
3) ingress at a port different from, but egress with, the foreground traffic.

To implement a configuration with background traffic, one or more ports need to be reserved for foreground traffic. Therefore, to implement:

Case 1 (different ingress and egress) – reserve 2 ports for each foreground traffic stream.

Cases 2 & 3 (shared ingress or egress) – reserve 1 port per foreground traffic stream.

B.3.4. Examples of Scalable Connection Configurations

In this section, several examples of scalable connection configurations are provided. In all of the examples, a switch with two 4-port modules is used (as in Figure B.4a), where all ports are of the same speed and connector type.

B.3.4.1. n-to-n Straight (Single Generator)

This configuration may be used for throughput as well as latency measurements. Suitable test configurations that employ serial traffic replication can be obtained as follows:

a) Foreground Traffic (Throughput Measurements)

For these tests, only a single chain starting from a single generator is needed. Figures B.4a and B.4b, provide adequate examples of this type of connection configuration.

b) Background Traffic (Latency Measurements)

In this example, the background traffic does not use the input/output ports of the foreground traffic. As shown in Figure B.6, generator/analyzer G_1 is used for background traffic while generator/analyzer G_2 is used for foreground traffic.

The background traffic passes through each port not assigned to the foreground traffic. Inter-module external connections are established by following the guidelines described in Appendix B.3.1, resulting in wires W_1, W_2, and LB_1. The VCC chain of the background traffic corresponds to the wire ordering W_1W_2. So the VCC chain is given by:

$$G_1\text{-}V_1\text{-}W_1\text{-}V_2\text{-}W_2\text{-}V_3\text{-}LB_1\text{-}V_3\text{-}W_2\text{-}V_2\text{-}W_1\text{-}V_1\text{-}G_1, \text{ where,}$$

by establishing exclusively inter-module VCCs: $V_1 = (P_{12}, P_{22})$, $V_2 = (P_{13}, P_{23})$, $V_3 = (P_{14}, P_{24})$.

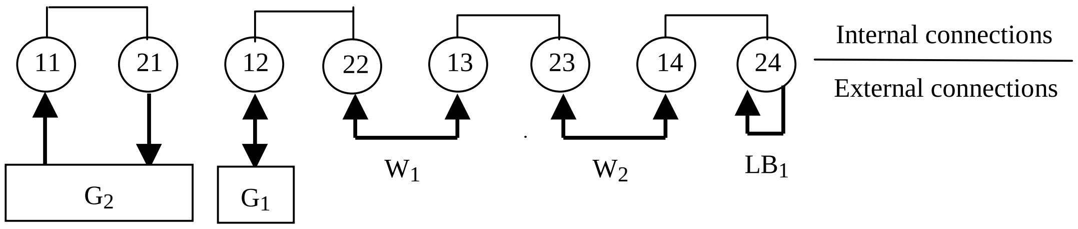


Figure B.6: The 6-to-6 Straight Configuration with One Generator using Inter-Module VCCs, where the Foreground Traffic does not Share any port with the Background Traffic.

B.3.4.2. n-to-n Straight (r Generators)

Foreground Traffic

To realize an n-to-n straight configuration with r generators, we need to design r VCC chains. Of the n ports, r ports are used as the source/destination of these chains. The remaining ports are connected among themselves and their wires are divided in subgroups among the generators ensuring an odd number of ports in each subgroup.

As an example, consider the 8-port switch again, with r = 3 generators. In dividing the available ports in three subgroups the goal is to have an odd number of ports in each subgroup. The partitioning of ports into subgroups results in one subgroup with three ports and two with one port each. The three ports subgroup is used by the first generator. Note that the wiring configurations of the other two subgroups consist only of a single loopback. Figure B.7 illustrates the implementation of the VCC chains for this case. First we select the source and destination ports. Let, P_{11} be the input/output port for the first chain, P_{21} be the input/output port for the second chain, and P_{12} be the input/output port for the third chain. Using only intra-module VCCs the following VCC chains are obtained:

$$G_1\text{-}V_1\text{-}W_1\text{-}LB_1\text{-}W_1\text{-}V_1\text{-}G_1,$$

$$G_2\text{-}V_2\text{-} LB_2\text{-}V_2\text{-}G_2, \text{ and}$$

G_3-V_3- LB_3-V_3-G_3, where

$V_1 = (P_{11}, P_{13})$, $V_2 = (P_{21}, P_{24})$ and $V_3 = (P_{12}, P_{14})$.

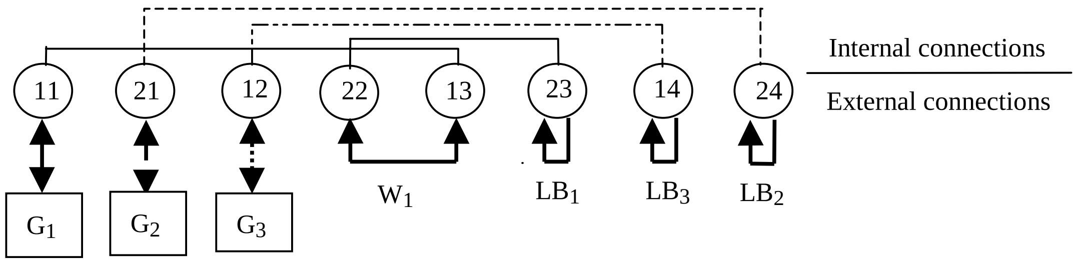


Figure B.7: Implementation of the 8-to-8 Straight Configuration with 3 Generators using Intra-module VCCs.

B.3.4.3. n-to-m Partial Cross (r Generators)

This configuration has m*r VCC chains starting from r generators**,** where each generator originates m VCC chains. Each chain has a load of 1/mth of the traffic egressing from the generator. Each intermediate wire has exactly m of these streams flowing through it. There are r subgroups and the switch ports, other than those reserved to be connected to the r generators, are divided among them.

a) Foreground Traffic: 8-to-2 (m=2) Partial Cross Configuration with 2 Generators (r=2).

For the configuration example shown in Figure B.8, there are two subgroups (one for each generator) with three ports each. The following intra-module VCC chains form the required test configuration:

G_1-V_1-W_1-V_2-LB_1-V_2-W_1-V_1-G_1,

G_2-V_3-W_2-V_4-LB_2-V_4-W_2-V_3-G_2,

G_1-V_5-W_1-V_6-LB_1-V_6-W_1-V_5-G_1, and

G_2-V_7-W_2-V_8-LB_2-V_8-W_2-V_7-G_2, where

$V_1 = (P_{11}, P_{13})$, $V_2 = (P_{21}, P_{23})$, $V_3 = (P_{12}, P_{14})$, $V_4 = (P_{22}, P_{24})$, $V_5 = (P_{11}, P_{14})$, $V_6 = (P_{22}, P_{24})$,

$V_7 = (P_{12}, P_{13})$, and $V_8 = (P_{21}, P_{23})$.

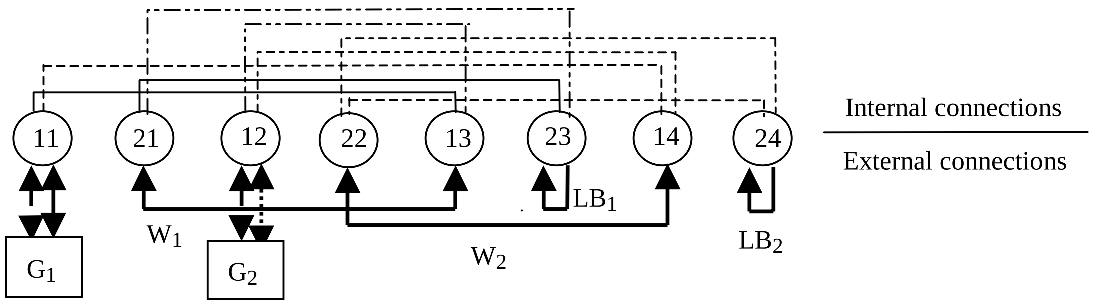


Figure B.8: Implementation of 8-to-2 Partial Cross Configuration with 2 Generators for Foreground Traffic using Intra-Module VCCs.

b) Foreground Traffic: 8-to-3 Partial Cross with One Generator

This case is illustrated in Figure B.9. There is only one subgroup composed of wires W1, W2, W3. The three VCC chains required for this test configuration (each constructed from a different wire ordering, with inter-module VCCs) are:

generator-V_1-W_1-V_2-W_2-V_3-W_3-V_4-LB_1-V_4-W_3-V_3-W_2-V_2-W_1-V_1-analyzer,

generator-V_5-W_2-V_6-W_3-V_7-W_1-V_8-LB_1-V_8-W_1-V_7-W_3-V_6-W_2-V_5-analyzer, and

generator-V_9-W_3-V_{10}-W_1-V_{11}-W_2-V_{12}-LB_1-V_{12}-W_2-V_{11}-W_1-V_{10}-W_3-V_9-analyzer,

where for tests involving exclusively inter-module traffic, the VCCs are:

$V_1 = (P_{11}, P_{21})$, $V_2 = (P_{12}, P_{22})$, $V_3 = (P_{13}, P_{23})$, $V_4 = (P_{14}, P_{24})$, $V_5 = (P_{11}, P_{22})$, $V_6 = (P_{13}, P_{23})$, $V_7 = (P_{14}, P_{21})$, $V_8 = (P_{12}, P_{24})$, $V_9 = (P_{11}, P_{23})$, $V_{10} = (P_{14}, P_{21})$, $V_{11} = (P_{12}, P_{22})$, $V_{12} = (P_{13}, P_{24})$.

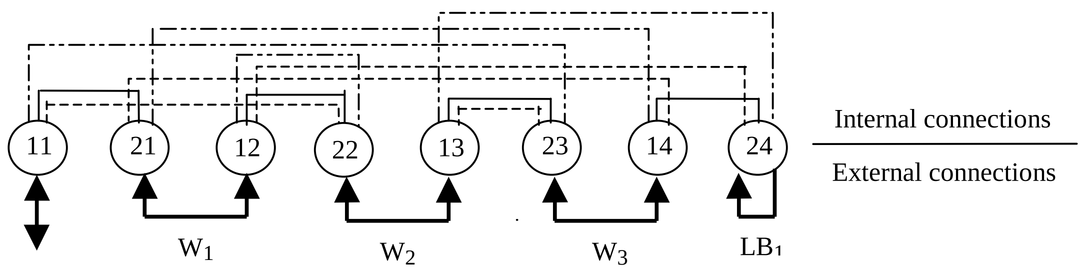


Figure B.9: 8-to-3 Partial Cross with One Generator Using Inter-Module VCCs

Appendix C: Statistical Analysis of ATM Performance Parameters

This appendix contains a variety of statistical procedures for use in performance testing. These procedures can be applied to estimate confidence intervals of the statistical parameters of interest (*i.e.*, mean, variance). The application of each of these procedures depends on the characteristics of the measured data. If the data are statistically independent or of large sample size, then the asymptotic procedure of Appendix C.1 applies. If the measured data are statistically independent, but the sample size is small, then the Bootstrap procedure described in Appendix C.2 applies. If the measured data are correlated and the sample size is not large, then the Jackknife procedure of Appendix C.3 applies. If the measured data are correlated and the sample size is too small, it is recommended that more data be collected. Figure C.1 summarizes the applicability of various methods. This doesn't imply that other appropriate statistical techniques are not to be used.

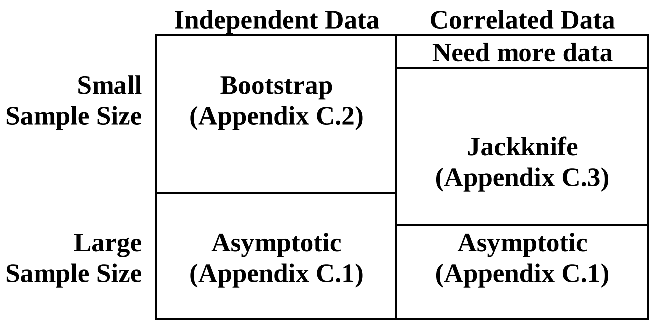


Figure C.1: Applicability of Various Methods

C.1. Confidence Interval Estimation for Large Sample Sizes: Asymptotic

Given the mean and the standard error, the users can compute a 100(1–α)-percent confidence interval as follows:

100(1–α)-percent confidence interval = (mean – z × standard error, mean + z × standard error)

Here, z is the (1–α/2)-quantile of the unit normal variate. For commonly used confidence levels, the quantile values are as follows:

Confidence	α	Quantile
90%	0.1	1.6448
95%	0.05	1.9600
99%	0.01	2.5758
99.9%	0.001	3.2905

The sample size can be chosen differently from its default value to obtain the desired confidence level.

C.2. Accuracy of Confidence Interval Estimation for Small Sample Sizes

Suppose that $X=(x_1, \ldots, x_N)$ is an independent, identically distributed (i.i.d.) sample from a population with distribution F. The sample is studied to estimate a certain parameter, $\Psi(F)$, associated with the distribution, F, whose form is unknown. A statistic, $S=S(X)$, is used to estimate $\Psi(F)$ from the data. However, a measure of the statistical accuracy of the point estimator $S(X)$ is also desired. In other words, the deviation of $S(X)$ from $\Psi(F)$ must be statistically quantified to gauge how much importance to attach to $S(X)$. For example, the bias (also known as the "systematic error") and the variance (which is responsible for the "random error") of the estimator, S, are of interest and are defined as follows:

$Bias_F(S) = \mathrm{E}_F[S(X)] - \Psi(F)$, and

$Var_F(S) = \mathrm{E}_F[S(X)^2] - (\mathrm{E}_F[S(X)])^2$.

where E_F denotes expectation under distribution F. The quantity $S(X) - \Psi(F)$, *i.e.*, the estimator minus the estimand, represents the error in estimating $\Psi(F)$ by $S(X)$.

C.2.1. Confidence Intervals Based on Asymptotic Normality

If the bias, $Bias_F(S)$, is negligible (compared to the square root of the variance $Var_F(S)$), a $(1 - \alpha)100\%$ confidence interval for $\Psi(F)$ will be the following:

$$\left[S(X) - z \times \sqrt{Var_F(S)},\ S(X) + z \times \sqrt{Var_F(S)}\right].$$

where $z=z(1 - \alpha/2)$ is the $1 - \alpha/2$ quantile of the standard Gaussian distribution.

If $Bias_F(S)$ is not negligible, the confidence interval will be the following:

$$\left[S(X) - Bias_F(S) - z \times \sqrt{Var_F(S)},\ S(X) - Bias_F(S) + z \times \sqrt{Var_F(S)}\right].$$

The above confidence intervals are based on the fact that the shape of the large sample distribution of $S(X) - \Psi(F)$ is the bell-shaped Gaussian. To formulate these confidence intervals, $Bias_F(S)$ and $Var_F(S)$ are required.

C.2.2. Confidence Intervals Based on Bootstrap

This section introduces the non-parametric bootstrap technique [7], which obtains confidence intervals for a wide variety of ATM performance parameter statistics in general situations. The bootstrap is a powerful technique for assessing the accuracy of a parameter estimator in small sample size situations. For example, the bootstrap technique and a set of sample values can be used to estimate unknown parameters of a random process.

The basic idea of bootstrap methods is to use the sample data to compute a statistic and to estimate its sampling distribution without using "asymptotic" results. In many performance testing applications, it cannot always be assumed that the size of the available set of sample values is large enough that "asymptotic" results apply. Limitations due to time constraints and cost often allow only small portions of stationary data to be considered. In addition, random processes of interest may not be stationary. Thus, often in practice, methods depending on large sample sizes are not accurate.

Bootstrap methods work by repeatedly drawing a large number of random samples from the original sample. The statistic is then computed for each of the pseudo samples, and the resulting empirical distribution is an estimate of the sampling distribution. In other words, the original sample is treated as a surrogate for the population.

Formally, let $X=\{x_1, x_2, ..., x_N\}$ be a random sample (N independent and identically distributed random variables) from an unknown distribution, F. Let μ denote the desired unknown parameter of F, $\Psi(F)$, such as its mean. We wish to find an estimator and a (1-α)100% confidence interval for the mean μ. Usually, we estimate μ using the sample mean, as follows:

$$\widehat{\mu} = \frac{x_1 + x_2 + ... + x_N}{N}.$$

A confidence interval for μ can be found by determining the distribution of $\widehat{\mu}$ (over repeated samples of size N with replacement from the underlying distribution, F), and finding values for $\widehat{\mu}_L$ and $\widehat{\mu}_U$ such that:

$$P[\widehat{\mu}_L \le \mu \le \widehat{\mu}_U] = 1 - \alpha.$$

When N is large, the distribution of $\widehat{\mu}$ could be approximated by the Gaussian distribution, as per the central limit theorem, but such an approximation may not be valid in applications where N is small.

Bootstrap methods suggest that we assume that the random sample $X=\{x_1, ..., x_N\}$ itself constitutes the underlying distribution; then by resampling from X many times and computing $\widehat{\mu}$ for each of these samples, we get a bootstrap distribution, $\widehat{F}$, for $\widehat{\mu}$ that approximates the actual distribution of $\widehat{\mu}$, and from which a confidence interval for μ is estimated. Figure C.2 illustrates one bootstrap technique and the relations between its steps. These steps are detailed in Appendix C.2.2.1.

In general, for any statistic, $S(X)$, having found the quantiles (at least approximately), we can calculate the (1-α)100% bootstrap confidence intervals for $\Psi(F)$, which is given by the α/2 and (1-α/2) quantiles of distribution $\widehat{F}$.

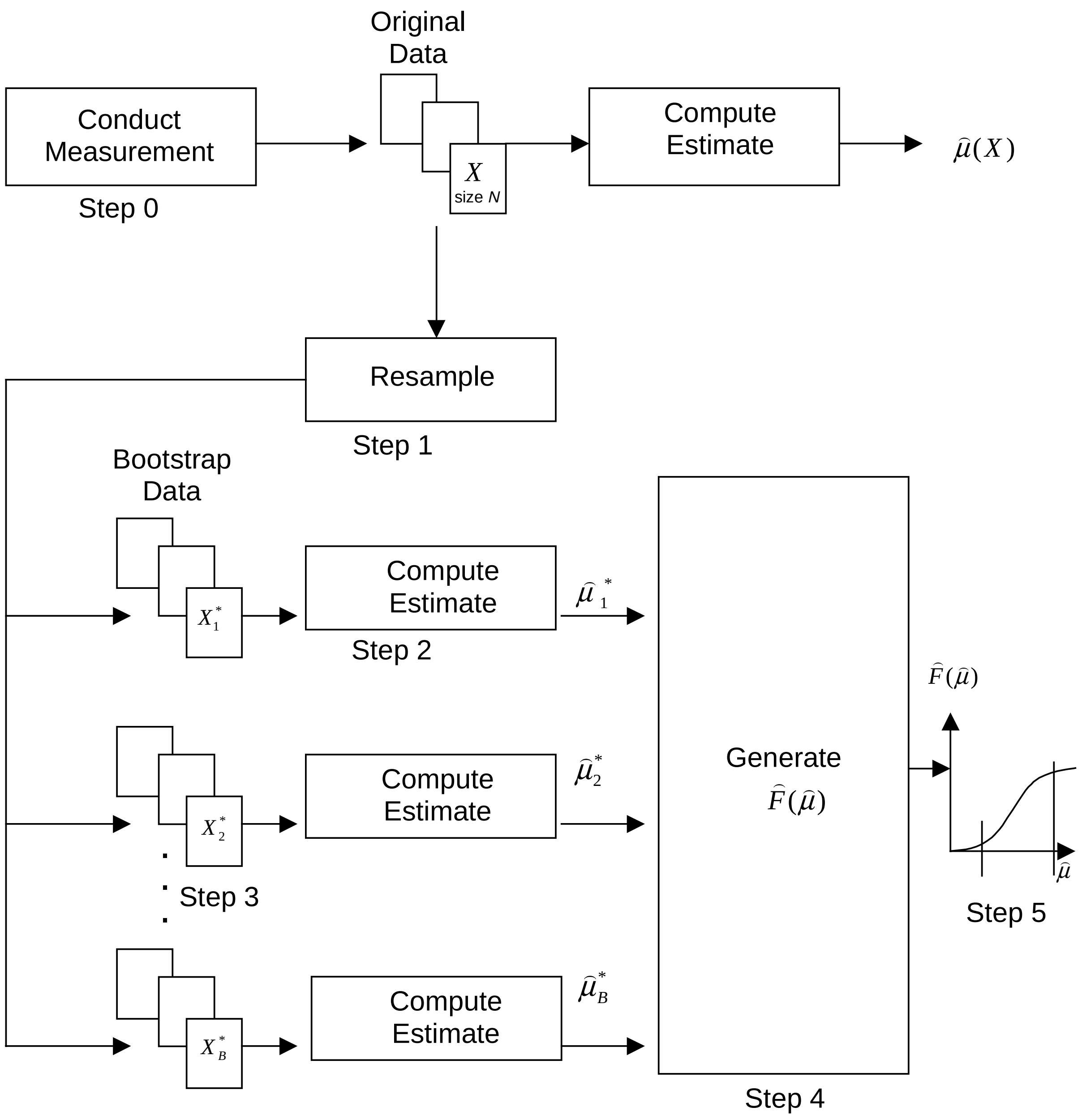


Figure C.2: Bootstrap technique for estimating a distribution

C.2.2.1. Bootstrap Steps for Calculating a Confidence Interval for the Mean

Step 0: Experiment. Suppose the original sample is $X = \{x_1, ..., x_N\}$

Step 1: Resampling. Draw a random sample of "N" values, with replacement, from X. This produces a bootstrap resample y (say X_1^*).

Step 2: Calculation of the bootstrap estimate. Calculate the mean for y (say, $\hat{\mu}_1^*$).

Step 3: Repetition. Repeat steps 1 and 2 many times (*e.g.*, 1000) to obtain B bootstrap estimates, $\hat{\mu}_1^*$, $\hat{\mu}_2^*$, …, $\hat{\mu}_B^*$.

Step 4: Approximation of the distribution of the mean. Sort the bootstrap estimates into increasing order, $\hat{\mu}_{(1)}^*$, $\hat{\mu}_{(2)}^*$, …, $\hat{\mu}_{(B)}^*$.

Step 5: Confidence interval using the percentile method. The desired (1- α) 100% bootstrap confidence interval for the mean is [$\hat{\mu}_{(q_1)}^*$, $\hat{\mu}_{(q_2)}^*$], where $q_1 = \lfloor B\alpha/2 \rfloor$, $q_2 = B\text{-}q_1+1$, and $\lfloor B\alpha/2 \rfloor$ is the integer part of $B\alpha/2$.

With minor modifications, these steps can also be used to calculate the confidence interval for the variance.

C.2.2.1.1. Example

Step 0: X = {4.86, 13.17, 6.06, 15.79, 10.20, 12.81, 14.10, -2.41, 15.77, 9.11}, $\hat{\mu}$ = 9.946.

Step 1: X_1^* = {9.11, 9.11, 6.06, 13.17, 10.20, -2.41, 4.86, 12.81, -2.41, 4.86}.

Step 2: $\hat{\mu}_1^*$ = 6.54.

Step 3: Repeat steps 1 and 2 to obtain n bootstrap estimates $\hat{\mu}_2^*$, …, $\hat{\mu}_B^*$ (B = 1000).

Step 4: $\hat{\mu}_{(1)}^* \le \hat{\mu}_{(2)}^* \le \ldots \le \hat{\mu}_{(1000)}^*$. For example, we might get 3.39, 3.48, 4.46, …, 8.86, 8.88, 8.89, …, 10.07, 10.08, …, 14.46, 14.53, 14.66.

Step 5: The desired (1- α) 100% bootstrap confidence interval for the mean is [$\hat{\mu}_{(q_1)}^*$, $\hat{\mu}_{(q_2)}^*$], where $q_1 = \lfloor B\alpha/2 \rfloor$, $q_2 = B\text{-}q_1+1$, $\lfloor B\alpha/2 \rfloor$ is the integer part of $B\alpha/2$, B = 1000, q_1 = 25, q_2 = 976, and the 95% confidence interval is found to be [6.27, 13.19].

C.2.2.1.2. Comparison of Bootstrap and Asymptotic Methods

Table C.1 compares confidence interval bounds for the bootstrap and asymptotic methods over selected sample sizes between N = 2 and N = 100. The bootstrap confidence interval for the mean is computed using the non-parametric bootstrap and percentile method described in C.2.2.1. The asymptotic confidence interval for the mean is computed using the standard small sample method based on the Student's *t* distribution. *E.g.*, for N = 10 and α = 0.95, the asymptotic confidence interval for the mean is:

$$[\hat{\mu} \pm t_{N-1,1-\alpha/2}\,(\hat{\sigma}/\sqrt{N})] \;=\; [0.827 \pm (2.262 \times 0.253)] \;=\; [0.255,\, 1.399]\,.$$

The δ_{bound} columns are normalized differences expressed as percentages. The δ_{bound} is negative when the asymptotic bound is closer to the sample mean than the bootstrap bound. *E.g.,* for N = 25, the asymptotic confidence interval's lower bound is 5.278% farther from the sample mean than the bootstrap bound, and the asymptotic confidence interval's upper bound is 0.243% nearer the sample mean than the bootstrap bound.

Table C.1: An example comparing the confidence interval bounds for the bootstrap and asymptotic methods. The δ_{bound} is negative when the asymptotic bound is closer to the sample mean.

Sample size	**Sample mean**	**Bootstrap confidence interval**	**Asymptotic confidence interval**	$\delta_{lower\ bound}$ **%**	$\delta_{upper\ bound}$ **%**
2	0.654	[0.294, 1.014]	[-3.920, 5.228]	1433.333 %	415.582 %
3	0.498	[0.185, 1.014]	[-0.621, 1.617]	435.676	59.467
4	0.695	[0.240, 1.150]	[-0.164, 1.554]	168.333	35.130
5	0.895	[0.394, 1.395]	[0.093, 1.697]	76.396	21.649
6	0.798	[0.385, 1.215]	[0.142, 1.454]	63.117	19.671
7	1.049	[0.534, 1.687]	[0.241, 1.857]	54.869	10.077
8	0.956	[0.433, 1.543]	[0.244, 1.668]	43.649	8.101
9	0.889	[0.421, 1.416]	[0.257, 1.521]	38.955	7.415
10	0.827	[0.391, 1.344]	[0.255, 1.399]	34.783	4.092
11	0.817	[0.431, 1.320]	[0.307, 1.327]	28.770	0.530
12	0.891	[0.500, 1.338]	[0.402, 1.380]	19.600	3.139
13	0.967	[0.582, 1.368]	[0.494, 1.440]	15.120	5.263
14	0.964	[0.576, 1.367]	[0.530, 1.398]	7.986	2.268
15	0.952	[0.610, 1.318]	[0.549, 1.355]	10.000	2.807
16	0.902	[0.585, 1.269]	[0.512, 1.292]	12.479	1.812
17	0.904	[0.604, 1.281]	[0.539, 1.269]	10.762	-0.937
18	0.948	[0.639, 1.280]	[0.594, 1.302]	7.042	1.719
19	0.994	[0.690, 1.297]	[0.647, 1.341]	6.232	3.392
20	1.017	[0.737, 1.300]	[0.686, 1.348]	6.920	3.692
25	0.957	[0.720, 1.235]	[0.682, 1.232]	5.278	-0.243
30	1.033	[0.772, 1.283]	[0.755, 1.311]	2.202	2.182
35	1.045	[0.774, 1.344]	[0.743, 1.347]	4.005	0.223
40	1.106	[0.842, 1.399]	[0.816, 1.396]	3.088	-0.214
50	1.021	[0.802, 1.273]	[0.774, 1.268]	3.491	-0.393
60	1.024	[0.805, 1.247]	[0.800, 1.248]	0.621	0.080
70	0.964	[0.766, 1.184]	[0.756, 1.172]	1.305	-1.014
80	0.905	[0.739, 1.099]	[0.719, 1.091]	2.706	-0.728
90	0.901	[0.737, 1.072]	[0.725, 1.077]	1.628	0.466
100	0.915	[0.750, 1.084]	[0.746, 1.084]	0.533	0.000

Figure C.3 and Table C.1 show that the bootstrap method produces a more compact confidence interval for small i.i.d. sample sizes (*e.g.,* N < 50). Confidence intervals are nearly identical for sample sizes larger than N = 50 in this example.

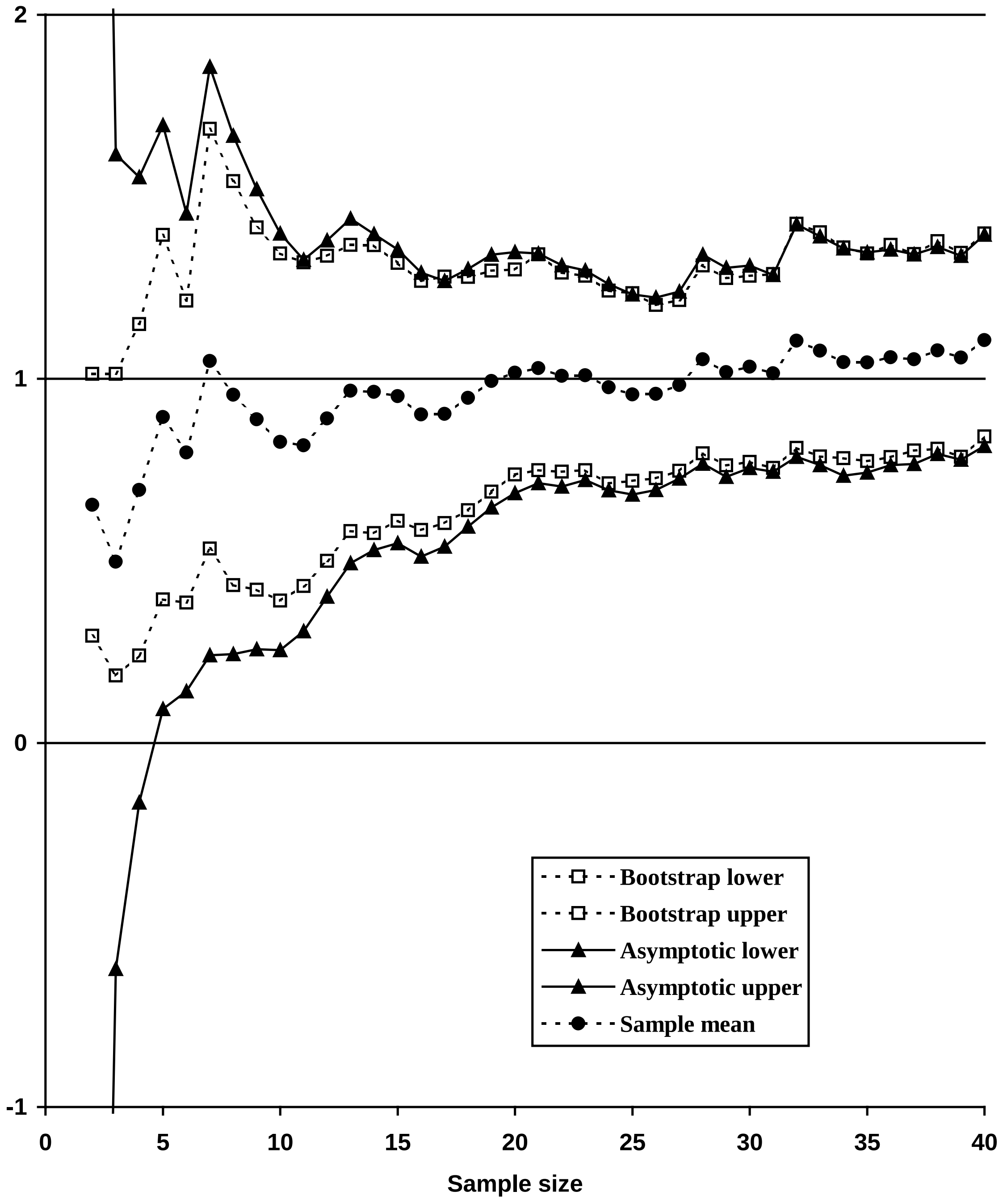


Figure C.3: A comparison of confidence intervals for the bootstrap and asymptotic methods for a range of sample sizes. Other realizations will produce different but generally consistent results

The bound between bootstrap (small i.i.d. sample) and asymptotic (large sample) methods on the left side of Figure C.1 is about N = 50 in this example. This is consistent with most textbook definitions of "small" sample sizes (N ≤ 50). One can use either method (bootstrap or asymptotic) for large samples, as both methods produce equivalent results.

The dataset used for the N = 100 sample is: {1.014, 0.294, 0.185, 1.286, 1.694, 0.316, 2.552, 0.309, 0.346, 0.269, 0.718, 1.706, 1.879, 0.932, 0.778, 0.151, 0.935, 1.708, 1.819, 1.447, 1.272, 0.562, 1.048, 0.223, 0.480, 1.010, 1.608, 2.960, 0.032, 1.469, 0.472, 3.884, 0.184, 0.013, 1.030, 1.532, 0.870, 1.979, 0.294, 2.982, 0.268, 0.236, 0.586, 1.880, 1.163, 0.459, 0.444, 0.178, 1.517, 0.094, 3.655, 0.140, 0.721, 0.590, 1.224, 0.727, 1.008, 0.774, 0.731, 0.830, 0.213, 0.169, 0.958, 0.509, 2.772, 0.243, 0.062, 0.309, 0.103, 0.695, 0.442, 0.907, 0.454, 0.519, 0.500, 0.410, 0.361, 0.439, 0.151, 0.710, 0.539, 0.679, 0.188, 2.905, 0.305, 0.659, 1.317, 0.015, 0.266, 1.851, 1.519, 0.517, 0.361, 1.141, 3.160, 0.104, 0.170, 0.181, 2.268, 0.980}.

C.3. Accuracy of Confidence Interval Estimation for Non-I.I.D. Data

The assumption of independent, identically distributed (i.i.d.) data breaks down either because the data are not independent or because they are not identically distributed or both. Let $X = \{x_1, \ldots, x_N\}$ be the random sample from the population. This section considers the case where the x_i data are dependent. The difficulty in this case is solved by using the Jackknife technique, introduced in this section. The Jackknife technique is relatively insensitive to the unknown distribution of X. A general discussion of this technique is provided in [16].

In the general Jackknife technique, B subsamples, $X^*_{(1)}$, …, $X^*_{(B)}$, are randomly chosen from among all the possible subsamples of size b from X. The number of possible subsamples is $N!/b!(N-b)!$ and a smaller number, B, is chosen to be included in the Jackknife technique. The ith subsample is $X^*_{(i)}$. The next step is to evaluate the statistic S over each chosen subsample, creating the pseudo-replications $S(X^*_{(1)})$, …, $S(X^*_{(B)})$. These subsamples are actually samples (of smaller size) from the true distribution, F, whereas the bootstrap resamples are samples from an estimator of F.

We now demonstrate the Jackknife technique for the case of estimating the variance and its confidence interval. Estimates and confidence intervals for the mean are insensitive to non-i.i.d. data (asymptotic normal or bootstrap techniques can be applied, even for non-i.i.d. data).

C.3.1. Confidence Intervals for the Variance

If there is correlation in the measured data, the standard statistical methods of obtaining an estimate of the variance and generating a confidence interval of the variance cannot be applied. The Jackknife technique solves this problem.

Let $\hat{\sigma}^2$ be the unbiased sample variance of X. That is,
$\hat{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\mu})^2 = \frac{1}{N-1}\sum_{i=1}^{N} x_i^2 - \frac{N}{N-1}\hat{\mu}^2$, where $\hat{\mu}$ is the sample mean of X: $\frac{1}{N}\sum_{i=1}^{N} x_i$.

Let $\hat{\sigma}_j^2$ (j=1, …, N) be the unbiased sample variance of X with the jth observation removed. That is, $\hat{\sigma}_j^2 = \frac{1}{N-2}\left(\sum_{i\neq j} x_i^2\right) - \frac{N-1}{N-2}\hat{\mu}_j^2$, where $\hat{\mu}_j$ is the sample mean of X with the jth observation removed: $\frac{1}{N-1}\sum_{i\neq j} x_i$.

The Jackknife case *b*=*N*-1, leads to *N* subsamples. This is known as the “ungrouped” or Quenouille method. Thus, $b = N-1$, $B = N$, and $S(X^*_{(j)}) = \hat{\sigma}^2_j$.

Let $y_j = N\hat{\sigma}^2 - (N-1)\hat{\sigma}^2_j$, where *j*=1, …, *N*. Note that E[y_j]=σ^2, where σ^2 is the variance of X, and $\hat{\sigma}^2$ is an unbiased point estimator of σ^2. Therefore, the distribution of y_j is centered about σ^2.

Let $\bar{y}$ be the unbiased sample mean of y_j, and $\hat{\sigma}^2_y$ be the sample variance;

$$\bar{y} = \frac{1}{N}\sum_{j=1}^{N} y_j \text{, and } \hat{\sigma}^2_y = \frac{1}{N-1}\sum_{j=1}^{N}\left(y_j - \bar{y}\right)^2 .$$

Given these definitions, $\dfrac{\bar{y}-\sigma^2}{\hat{\sigma}_y/\sqrt{N}}$ has approximately a *t*-distribution with *N*-1 degrees of freedom. Thus, the (1-α)100% confidence interval for σ^2 is:

$$\left[\bar{y} - t_{N-1,(1-\alpha/2)}\hat{\sigma}_y/\sqrt{N},\ \bar{y} + t_{N-1,(1-\alpha/2)}\hat{\sigma}_y/\sqrt{N}\right].$$

Figure C.4 illustrates this Jackknife technique and the relations between its steps. These steps are detailed in Appendix C.3.1.1.

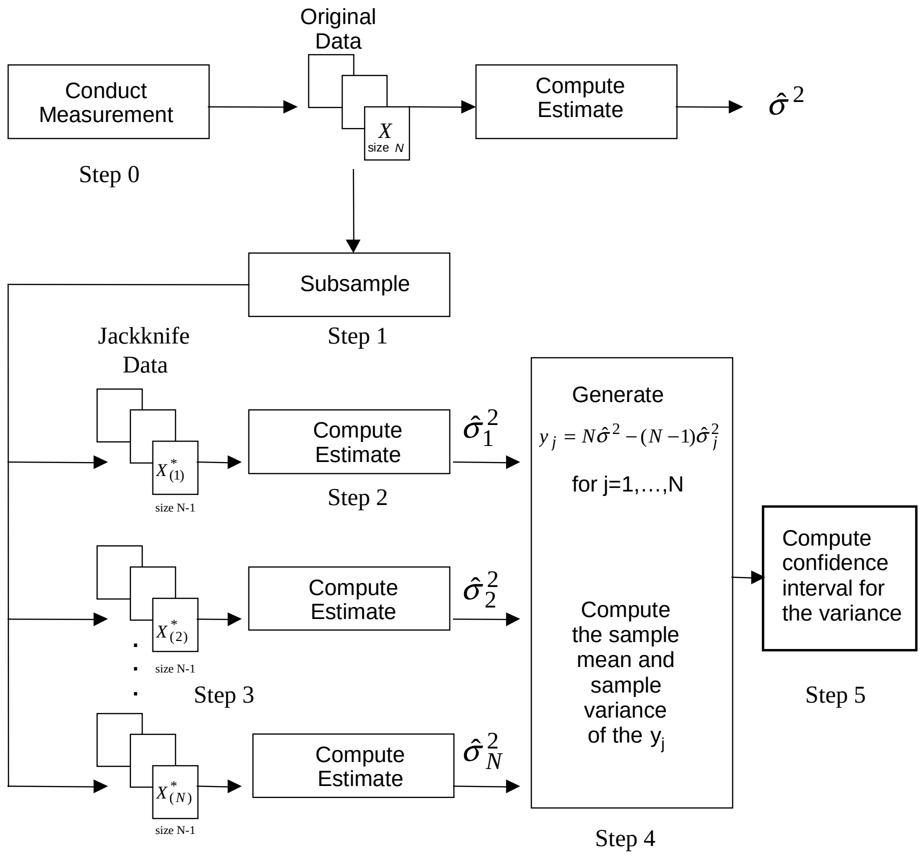


Figure C.4: Jackknife Technique for Estimating the Variance.

C.3.1.1. Jackknife Steps for Calculating a Variance Estimate and Confidence Interval

Step 0: Experiment. Suppose the original sample is $X = \{x_1, ..., x_N\}$. The unbiased sample variance of X is $\hat{\sigma}^2$.

Step 1: Subsampling. Remove observation 1 from X. This subsample (with N-1 observations) is denoted $X^*_{(1)}$.

Step 2: Calculation of the variance estimate. Calculate the unbiased sample variance of the subsample $X^*_{(1)}$. This is denoted $\hat{\sigma}_1^2$.

Step 3: Repetition. Repeat steps 1 and 2 for the other N-1 subsamples that are obtained by removing observation j (j=2,…,N) from X at each repetition. Steps 1-3 provide N variance estimates, $\hat{\sigma}_1^2$, $\hat{\sigma}_2^2$, …, $\hat{\sigma}_N^2$.

Step 4: Define pseudovalues, and compute their sample statistics. Define N pseudovalues, as $y_j = N\hat{\sigma}^2 - (N-1)\hat{\sigma}_j^2$ for j=1,…,N. Compute the sample mean of the y_j, as $\bar{y} = \frac{1}{N}\sum_{j=1}^{N} y_j$. Compute the unbiased sample variance of the y_j, as $\hat{\sigma}_y^2 = \frac{1}{N-1}\sum_{j=1}^{N}(y_j - \bar{y})^2$.

Step 5: Confidence interval using the ungrouped jackknife method. The desired (1-α)100% jackknife confidence interval for the population variance is $[\bar{y} \pm t_{N-1,(1-\alpha/2)}\hat{\sigma}_y/\sqrt{N}]$, where $t_{N-1,(1-\alpha/2)}$ is the critical value that leaves (1-α/2)100% probability in the upper tail of the Student's t distribution with N-1 degrees of freedom.

The sample variances in steps 0, 2, 3 and 4 can be computed as the data are collected (sampled or subsampled). Multiple passes through the data are not required.

C.3.1.1.1. Example

Step 0: X = {0.36292, 0.74519, 0.83106, …, 0.23004}. N = 30 in this example, and $\hat{\sigma}^2$ = 0.07205 .

Step 1: $X^*_{(1)}$ = {0.74519, 0.83106, …, 0.23004}. There are N-1 = 29 observations in this subsample.

Step 2: $\hat{\sigma}_1^2$ = 0.07438 .

Step 3: Repeat steps 1 and 2 to obtain a total of N = 30 subsamples, and 30 sample variance estimates: $\hat{\sigma}_1^2$ = 0.07438, $\hat{\sigma}_2^2$ = 0.07130, …, $\hat{\sigma}_N^2$ = 0.07292 .

Step 4: $y_1 = N\hat{\sigma}^2 - (N-1)\hat{\sigma}_1^2$ = 0.00466 . Similarly, compute y_j, j=2,…,30. Then compute $\bar{y}$ = 0.07205 and $\hat{\sigma}_y^2$ = 0.00454, the sample mean and sample variance of the 30 y_j values.

Step 5: The desired 95% confidence interval for the variance is [0.04689, 0.09721]. In this example, the true population variance is 0.08333.

C.3.1.1.2. Comparison of Jackknife and Asymptotic Methods

Table C.2 compares confidence interval bounds for the jackknife and asymptotic methods for correlated sample data having first-order autocorrelation coefficients between ρ = 0.01 and ρ = 0.99 and a sample size of N = 30. The jackknife confidence interval for the variance is computed using the ungrouped (b=N-1) jackknife method described in Appendix C.3.1.1. The asymptotic confidence interval for the variance is computed using the standard i.i.d. method based on the chi-square distribution (or the modified chi-square for larger samples). *E.g.*, for N = 30, ρ = 0.50 and α = 0.95, the asymptotic confidence interval for the variance is:

$$\left[\frac{(N-1)S^2}{\chi^2_{\alpha/2,\,N-1}}, \frac{(N-1)S^2}{\chi^2_{1-\alpha/2,\,N-1}}\right] = [29\times1.750/45.772,\ 29\times1.750/16.047)] = [1.109, 3.163]\,.$$

The δ_{bound} columns are normalized differences expressed as percentages. The δ_{bound} is negative when the asymptotic bound is closer to the sample variance than the jackknife bound. While it appears that the asymptotic lower bound is superior, it is actually an underestimate because the estimator is not robust in small, correlated samples.

Table C.2. An example comparing the confidence interval bounds for the jackknife and asymptotic methods. The δ_{bound} is negative when the asymptotic bound is closer to the sample variance.

Correlation coefficient	**Sample variance**	**Jackknife confidence interval**	**Asymptotic confidence interval**	$\delta_{\text{lower bound}}$ **%**	$\delta_{\text{upper bound}}$ **%**
0.01	1.051	[0.573, 1.529]	[0.666, 1.899]	-16.230	24.199
0.05	1.080	[0.595, 1.565]	[0.684, 1.952]	-14.958	24.728
0.10	1.122	[0.624, 1.619]	[0.711, 2.027]	-13.942	25.201
0.15	1.170	[0.657, 1.683]	[0.741, 2.114]	-12.785	25.609
0.20	1.225	[0.693, 1.757]	[0.776, 2.214]	-11.977	26.010
0.25	1.288	[0.732, 1.844]	[0.816, 2.328]	-11.475	26.247
0.30	1.360	[0.776, 1.944]	[0.862, 2.457]	-11.082	26.389
0.35	1.441	[0.823, 2.058]	[0.913, 2.604]	-10.936	26.531
0.40	1.532	[0.874, 2.190]	[0.971, 2.769]	-11.098	26.438
0.45	1.635	[0.928, 2.342]	[1.036, 2.955]	-11.638	26.174
0.50	1.750	[0.984, 2.517]	[1.109, 3.163]	-12.703	25.665
0.55	1.880	[1.042, 2.718]	[1.191, 3.397]	-14.299	24.982
0.60	2.026	[1.103, 2.949]	[1.284, 3.662]	-16.410	24.178
0.65	2.193	[1.171, 3.214]	[1.389, 3.963]	-18.617	23.304
0.70	2.385	[1.254, 3.516]	[1.511, 4.311]	-20.494	22.611
0.75	2.611	[1.368, 3.854]	[1.654, 4.718]	-20.906	22.418
0.80	2.879	[1.541, 4.216]	[1.824, 5.202]	-18.365	23.387
0.85	3.196	[1.814, 4.579]	[2.025, 5.777]	-11.632	26.163
0.90	3.579	[2.201, 4.957]	[2.268, 6.468]	-3.044	30.482
0.95	4.234	[2.702, 5.766]	[2.683, 7.652]	0.703	32.709
0.99	5.097	[3.187, 7.007]	[3.229, 9.211]	-1.318	31.454

Figure C.5 shows that the jackknife method produces a confidence interval upper bound that is significantly closer to the sample variance for weakly and moderately dependent data (*e.g.*, $\rho < 0.9$) with N = 30. Differences are even larger when $\rho \geq 0.9$. As noted, the asymptotic lower bound is unreliable and should be avoided in small, correlated samples. If the measured data are correlated and if the sample size is very small (*e.g.*, N < 10), then no method can guarantee a realistic confidence interval for the variance. More data need to be collected.

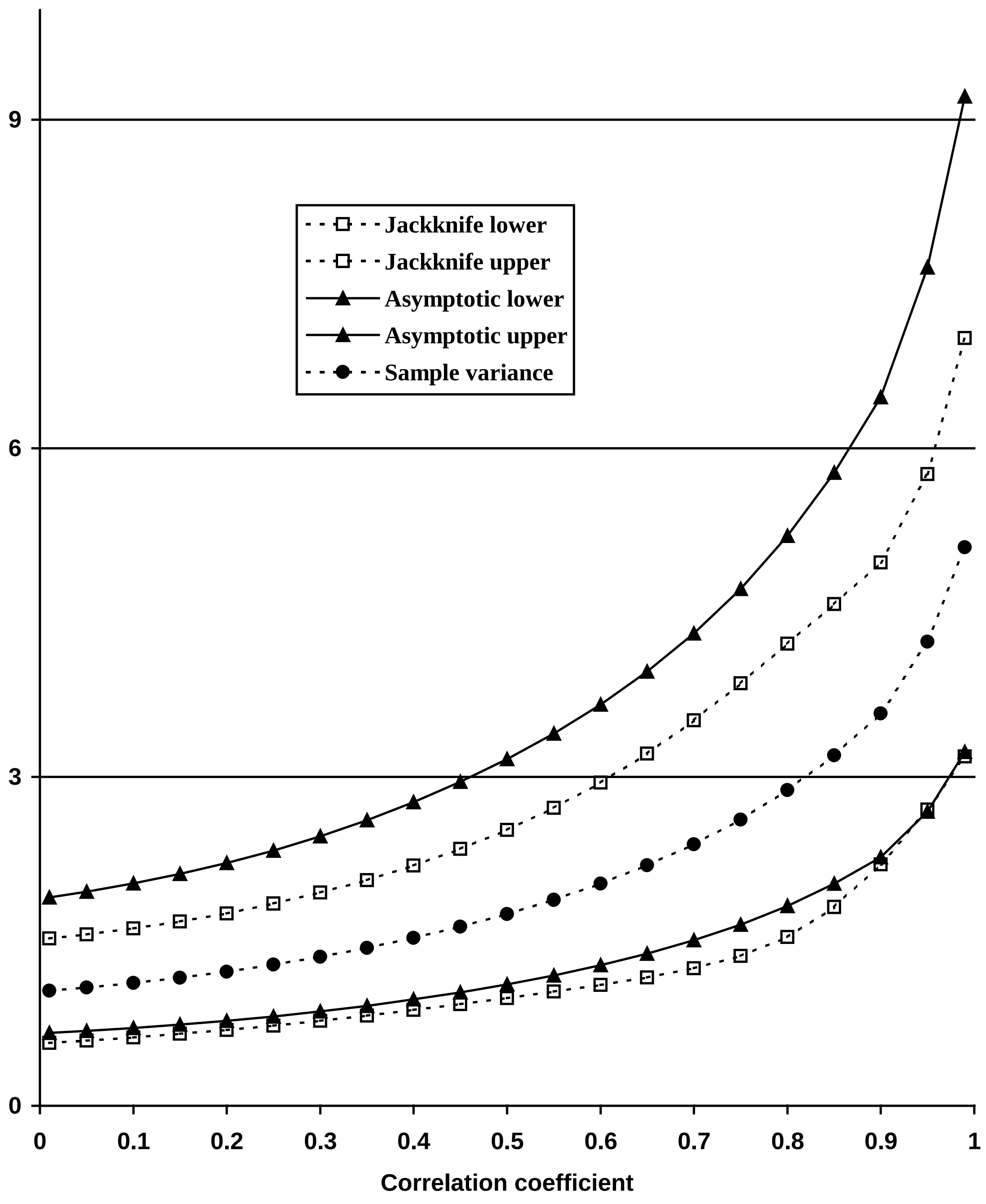


Figure C.5: A comparison of confidence intervals for the jackknife and asymptotic methods for a range of correlation coefficients. Other realizations will produce different but generally consistent results

In this example, the bound between jackknife (small correlated sample) and asymptotic (large sample) methods on the right side of Figure C.1 depends on both the sample size N and ρ. A sample with weak-to-moderate correlation (*e.g.*, $|\rho| < 0.5$) might require a sample size of 150 or more, while a sample with $|\rho|$ > 0.8 might require N ≈ 250. *E.g.,* large positive correlation means that a data point above the mean is likely to be followed by another data point above the mean, so one might see a disproportionately large number of "elevated" sample points in a small, correlated sample. This will bias the sample variance and the asymptotic confidence interval bounds if N is too small. One can use either method (jackknife or asymptotic) for large samples, as both methods produce equivalent results.

The dataset used for the ρ = 0.5, N = 30 example is: {1.14323, 0.77446, -0.89094, -0.09006, -0.27767, -0.22702, 0.89675, 0.05671, 1.92046, 2.40715, 1.70900, 0.53860, -0.32551, -0.62455, 0.33333, 2.46235, 2.28505, 0.49709, -0.51621, 0.96350, -1.17857, -2.23387, -1.95727, -1.89923, -1.66045, -0.05656, 0.60535, 2.00016, 1.35338, 1.01044}.

C.3.2. Generalized Jackknife

Let $x_{l,n}$ be the *n*th element of the *l*th subsample. Let $\hat{\mu}_l$ be the sample mean of the *l*th subsample. It is assumed here that the subsamples are approximately statistically independent. This will be true if, for example, $b << N$ and $B << N$. That is,

$$\hat{\mu}_l = \frac{1}{b}\sum_{n=1}^{b} x_{l,n} .$$

Let $\hat{\gamma}_j^2$ denote the mean square of the *j*th subsample. That is,

$$\hat{\gamma}_j^2 = \frac{1}{b}\sum_{n=1}^{b} x_{j,n}^2 .$$

Let $\hat{\sigma}_j^2$ (*j*=1, …, *B*) be the unbiased sample variance of *X* with the *j*th subsample removed ($X_{(j)}^*$). That is,

$$\hat{\sigma}_j^2 = \left(\frac{1}{B-1}\sum_{l\neq j, l=1}^{B}\hat{\gamma}_l^2\right) - \left(\frac{1}{B-1}\sum_{l\neq j, l=1}^{B}\hat{\mu}_l\right)^2 .$$

Let $\hat{\mu}$ denote the sample mean of *X* and $\hat{\sigma}^2$ denote the sample variance of X. That is,

$$\hat{\mu} = \frac{1}{B}\sum_{j=1}^{B}\hat{\mu}_j$$

$$\hat{\sigma}^2 = \frac{1}{B}\sum_{j=1}^{B}\hat{\gamma}_j^2 - \hat{\mu}^2$$

Let $y_j = B\hat{\sigma} \; - (B-1)\hat{\sigma}_j^2$, for $j = 1, \cdots, B$

Let $\bar{y}$ be the sample mean of y_j, and $\hat{\sigma}_y^2$ be the sample variance. That is,

$$\bar{y} = \frac{1}{B}\sum_{j=1}^{B} y_j$$

$\hat{\sigma}_y^2 = \frac{1}{B-1}\sum_{j=1}^{B}\left(y_j - \bar{y}\right)^2$, for $j = 1, \cdots, B$

Given these definitions, $\frac{\bar{y} - \sigma^2}{\hat{\sigma}_y / \sqrt{B}}$ has approximately a *t*-distribution with *B*-1 degrees of freedom. Thus, the (1-α)100% confidence interval for σ^2 is:

$$[\bar{y} - t_{B-1,1-\alpha/2}\hat{\sigma}_y / \sqrt{B},\ \bar{y} + t_{B-1,1-\alpha/2}\hat{\sigma}_y / \sqrt{B}].$$

Appendix D: In-service and Out-of-service Measurement of QoS Parameters

D.1. Introduction

Performance testing deals with in-service and out-of-service measurement of QoS parameters under different load condition profiles. In Appendix D.2, the out-of-service measurements are described, whereas in Appendix D.3, the in-service measurements are presented.

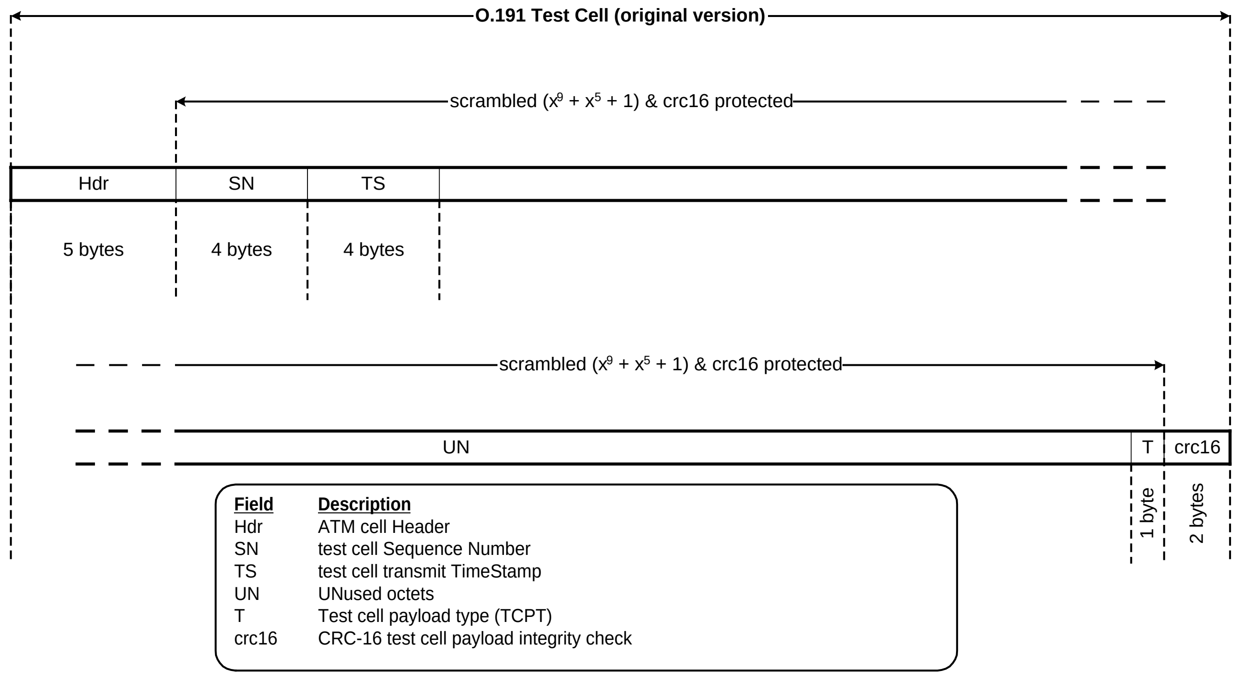
D.2. Out-of-service Measurement of QoS Parameters

D.2.1. Introduction

If the system under test can be taken out of service, then user cells can be replaced by test cells. These cells can be used together with a suitable measurement algorithm to measure the ATM performance parameters of the system under test.

D.2.2. Test Cell Format

The test cell format followed by ATM Forum is that provided in ITU-T Recommendation O.191 [9], shown in Figure D.1.


O.191 Test Cell (original version)
scrambled ($x^9 + x^5 + 1$) & crc16 protected
Hdr
SN
TS
5 bytes
4 bytes
4 bytes
scrambled ($x^9 + x^5 + 1$) & crc16 protected
UN
T
crc16
1 byte
2 bytes

Field	Description
Hdr	ATM cell Header
SN	test cell Sequence Number
TS	test cell transmit TimeStamp
UN	UNused octets
T	Test cell payload type (TCPT)
crc16	CRC-16 test cell payload integrity check

Figure D.1: ITU-T Recommendation O.191 Test Cell

The UN field (unused octets) can be used for proprietary purposes, provided this is indicated by the setting of the PPI (Proprietary Payload Indicator) bit in the T field (Test cell payload type).

D.2.3. Test Configuration

Since QoS parameters are defined as parameters that can directly be observed by users, the following definitions can be given:

- System under Test (SUT) is an ATM network or an ATM switch.
- Access for QoS measurement is the T or S reference point at the UNI.

Test cells, as used in this appendix, can be either the test cell format described in Appendix D.2.2, or those used in the AAL-5 test frames as described in Appendix D.2.5.

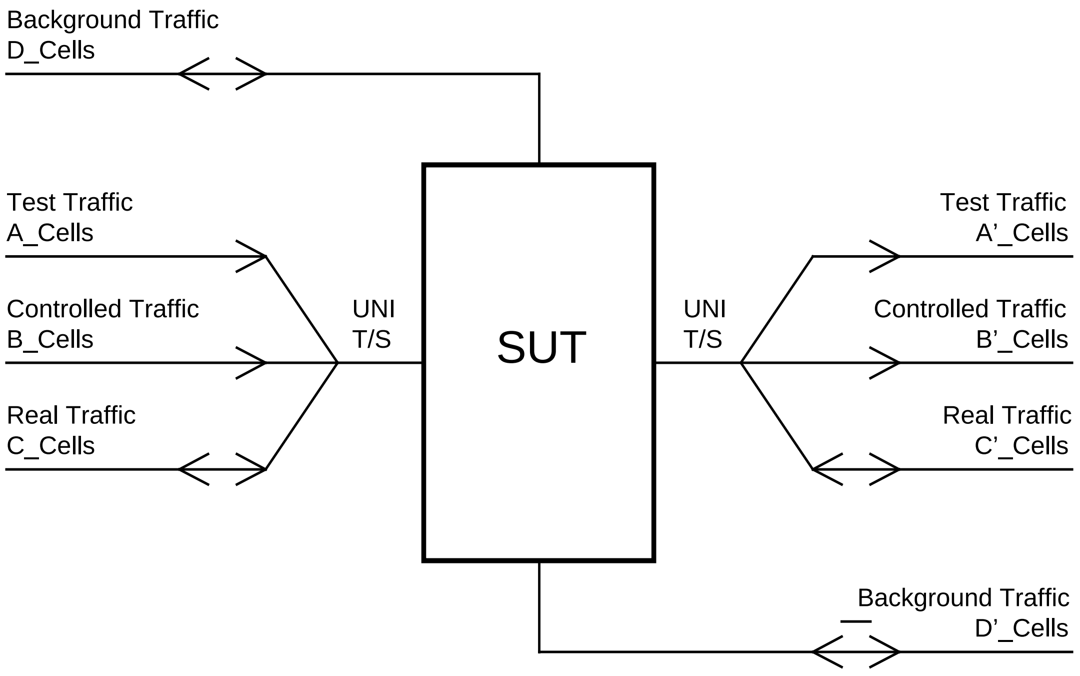


Figure D.2: QoS Measurement Arrangement

The SUT is loaded by the following cell traffic:

- Traffic at the Test Cell Input:
  - Test Traffic (A_Cells, unidirectional, called test cell stream) generated by the test equipment for the VP and/or VC used for testing purposes
  - Controlled Traffic (B_Cells, unidirectional) generated by the test equipment but for VPs/VCs not used for testing.
  - Real Traffic (C_Cells, bi-directional) generated/terminated by real subscribers connected to the UNI used for testing.
  - Traffic at the Test Cell Output:
  - Test Traffic (A'_Cells, unidirectional) terminated at the test equipment for the VP and/or VC used for testing purposes
  - Controlled Traffic (B'_Cells, unidirectional) terminated at the test equipment but for VPs/VCs not used for testing.
  - Real Traffic (C'_Cells, bi-directional) generated/terminated by real subscribers connected to the UNI used for testing.

- Background Traffic (D_Cells and D'_Cells, bi-directional): traffic generated by real subscribers not connected at the Test Cell Input or Test Cell Output.

In general, two types of test configurations exist:

1. SUT is fully under control by the tester. There is no real and no background traffic. Test traffic as well as the controlled traffic is generated by the test equipment. This configuration will allow reproducible test results to be obtained, because the QoS parameters very much depend on the overall traffic in the SUT.
2. SUT is loaded with real as well as background traffic that is out of control of the tester. It is difficult to get reproducible test results, mainly because the real and background traffic could lead to some overload conditions within the SUT. Therefore, it is necessary to measure not only the QoS parameters, but in parallel the load:
   - at the UNI of the Test Cell Input;
   - at the UNI of the Test Cell Output;
   - within the SUT.

D.2.3.1. Test Cell Input

The Test Cell Input is a T or S reference point. Two categories of cells can be generated:

A_Cells that form the test traffic. Each cell contains:
- VPI and VCI of a VP and VC established for testing purposes.
- Correct header (HEC).
- Cell payload containing an identification (*e.g.,* a cell sequence number, a time stamp).
- Payload has to be guarded by a CRC-16.

Test traffic conforms to the negotiated traffic contract. Note that any non-conformity could introduce cell loss and therefore a significant decrease of the QoS.

B_Cells that form controlled traffic. Each cell contains:
- VPI and VCI different from VP and VC used for testing purposes.
- Correct header (HEC).
- Cell payload.

D.2.3.2. Test Cell Output

The Test Cell Output is a T or S reference point. Since the QoS parameters are based on events of cells, actual measurements can only be performed above the physical layer.

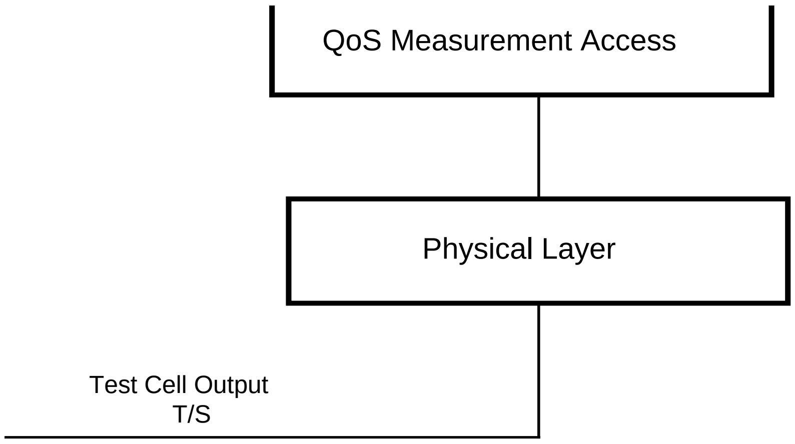


Figure D.3: QoS Measurement Access

The physical layer implemented in the test equipment may be considered as free of processing errors and without processing delay; or errors and delays in the physical layer should be taken into account in the calculation of the QoS parameter values.

QoS measurement access will get only valid cells from the physical layer because cells with a faulty HEC arriving at the Test Cell Output will get the header corrected or will be discarded. Therefore, it will not be possible to analyze faulty cells with a HEC error in the ATM layer.

At the QoS measurement access, the cell stream will be divided into the following two types of cells:

1. A'_Cells belong to the test traffic identified by the VPI or VPI/VCI. The following analysis will be done:
   - CRC of the payload:
     - ➢ If CRC is correct, the cell is assumed to be a valid test cell. In this case it is possible to analyze the cell identification in the payload. It shows that the cell is in the right sequence:
       - If yes, the cell is assumed to be a correct test cell (called a_cells).
       - If no, the cell is assumed to be a missequenced test cell (called b_cells).
     - ➢ If CRC is incorrect, the cell is assumed to be an errored test cell even if there is a possibility that the cell is a misinserted one (called c_cells).
2. B'_Cells do not belong to the test traffic but have to be taken into account (together with the real traffic) to calculate the total traffic at the Test Cell Output.

D.2.4. Analysis of the Test Cell Stream

D.2.4.1. Cell Error Ratio (CER)

A continuous test cell stream containing A_cells is sent at the Test Cell Input. At the Test Cell Output, the A'_cells are analyzed during a time interval of t1 (service dependent, the value is for further study). CER is calculated as follows:

$$CER = c / A'$$

where:

A’:	number of A’_cells (received test cells)
c:	number of errored A’_cells (c_cells: payload CRC is faulty)

This method will lead to incorrect results if misinserted cells with payload bit errors are received (overcount of c and A’).

D.2.4.2. Cell Loss Ratio (CLR)

A continuous test cell stream containing A_cells is sent at the Test Cell Input. At the Test Cell Output the A’_cells are analyzed during a time interval of t1 (service dependent, the value is for further study). CLR is calculated as follows:

$$CLR = (A - A') / A$$

where:

A:	number of sent A_cells (test cells)
A’:	number of received A’_cells (test cells)

This method will lead to incorrect results if misinserted cells with payload bit errors are received (overcount of A’). In case of misinserted cells > cell loss, CLR will be negative.

D.2.4.3. Cell Misinsertion Rate (CMR)

A continuous test cell stream containing only B_cells is sent at the Test Cell Input. At the Test Cell Output no A’_cells should arrive otherwise they are misinserted. Therefore A’_cells are counted during a time interval of t2 (service dependent, the value is for further study). CMR is calculated as follows:

$$CMR = A' / t2$$

where:

A’:	number of received A’_cells (test cells)

D.2.4.4. Cell Missequenced Ratio (CSR)

ITU-T Recommendation I.356 doesn’t define a cell missequenced ratio (CSR), because AAL controls missequencing of cells. Nevertheless, missequencing of cells is a faulty behavior of the ATM network that can be observed by the user. Therefore, an appropriate measuring method is recommended.

A continuous test cell stream containing A_cells is sent at the Test Cell Input. At the Test Cell Output, the A’_cells are analyzed during a time interval of t1 (service dependent, the value is for further study). CSR is calculated as follows:

$$CSR = b / A'$$

where:

A’:	number of received A’_cells (test cells)
b:	number of b_cells where the cell identification in the payload shows that the cell was overtaken by the previous cell (cell sequence error). Example:

Cell number :	1	3	4	2	5
Counter value b :	n	n	n	n+1	n+1

D.2.4.5. Measuring Cell Transfer Times

To be able to perform measurements of the cell transfer time, the equipment generating the test traffic and the equipment analyzing the received test traffic must be synchronized in the order of micro-seconds. Whereas CTD measurements require a synchronization in both frequency and absolute value (known offset), it maybe possible to measure CDV with only a synchronization in frequency. At the Test Cell Input, a time stamp (according to O.191) is inserted as cell identification in the A_Cells payload. At the Test Cell Output, only correct A'_Cells (a_cells and b_cells, no payload CRC fault) are analyzed. All the cell transfer time parameters are evaluated from the measurement of the Call Transfer Delay (CTD):

CTD = tr - ts

where:

tr:	receive time relative to the synchronized time reference when the cell reached the Test Cell Output side.
ts:	transmit time relative to the synchronized time reference when the cell left the Test Cell Input side.

Note: CTD is one of the cell transfer performance parameters. It very much depends on the cell traffic in the test traffic, the controlled traffic and the real traffic, as well as on the background traffic.

D.2.4.6. Mean Cell Transfer Delay (MCTD)

Mean Cell Transfer Delay is the arithmetic average (mean) of the CTD measured over the time period t1 (service dependent, the value is for further study).

MCTD(t1) = Scdt / a

where:

a:	number of received and correct A'_cells (no payload CRC fault: correct test cells)
Scdt:	Summation of the CTD (tr - ts) of all correct A'_cells.

D.2.4.7. Maximum Cell Transfer Delay

Maximum cell transfer delay is the maximum value of the CTD measured over the time period t1 (service dependent, the value is for further study) of all correct A'_cells.

MaxCTD = max (CTDi) i: 1, … n

where:

CTD_i:	the cell transfer delay of the i^{th} correct A'_cell received within t1.
n:	the total number of correct A'_cells received within t1.

This procedure is based on TM 4.1 [2] (Section 3.6.1.1) with the effect of alpha being neglected.

D.2.4.8. Peak-to-peak Two-point Cell Delay Variation

Peak-to-peak two-point cell delay variation is the maximum value of the CTD minus the minimum value of the CTD measured over the time period t1 (service dependent, the value is for further study) of all correct A'_cells.

peak-to-peak CDV = MaxCTD - MinCTD

where:

MinCTD = min (CTD_i) i: 1, …..n

with:

CTD_i: the cell transfer delay of the i^{th} correct A'_cell received within t1.
n: the total number of correct A'_cells received within t1.

D.2.5 AAL-5 Test Frame Format

The test frame is an AAL-5 frame made up of two kinds of instrumented test cells, Frame-body Test Cells and a single End-of-Frame (EOF) Test Cell.

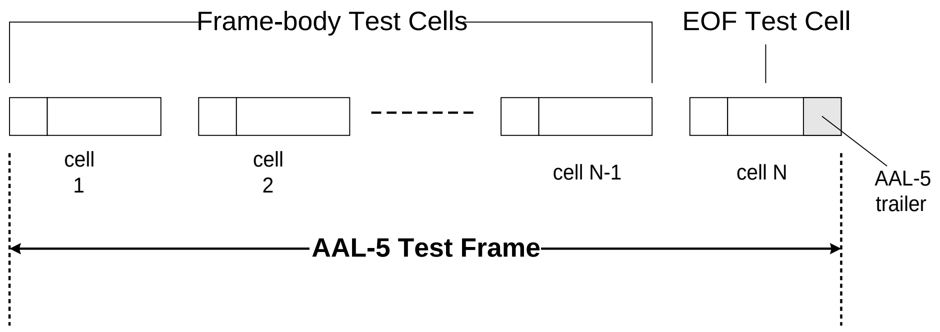


Figure D.4: AAL-5 Test Frame Format

For a frame of length N cells, the first N-1 cells are Frame-body test cells. The Nth cell is an EOF test cell that differs from the frame-body test cells only in that the last 8 octets of the cell are shifted left 8 octets to make room for the AAL-5 trailer.

A test frame that is only one cell long consists of a single EOF test cell.

By composing test frames of test cells it becomes possible to make simultaneous, correlated ATM layer and frame layer performance measurements on the same traffic.

Overall frame integrity is checked using the AAL-5 CRC-32 while individual cells retain the CRC-16 cell payload integrity check.

Due to the use of the EOF test cell, the AAL-5 test frame does not conform to ITU-T Recommendation O.191. The AAL-5 test frame allows the following five benefits:

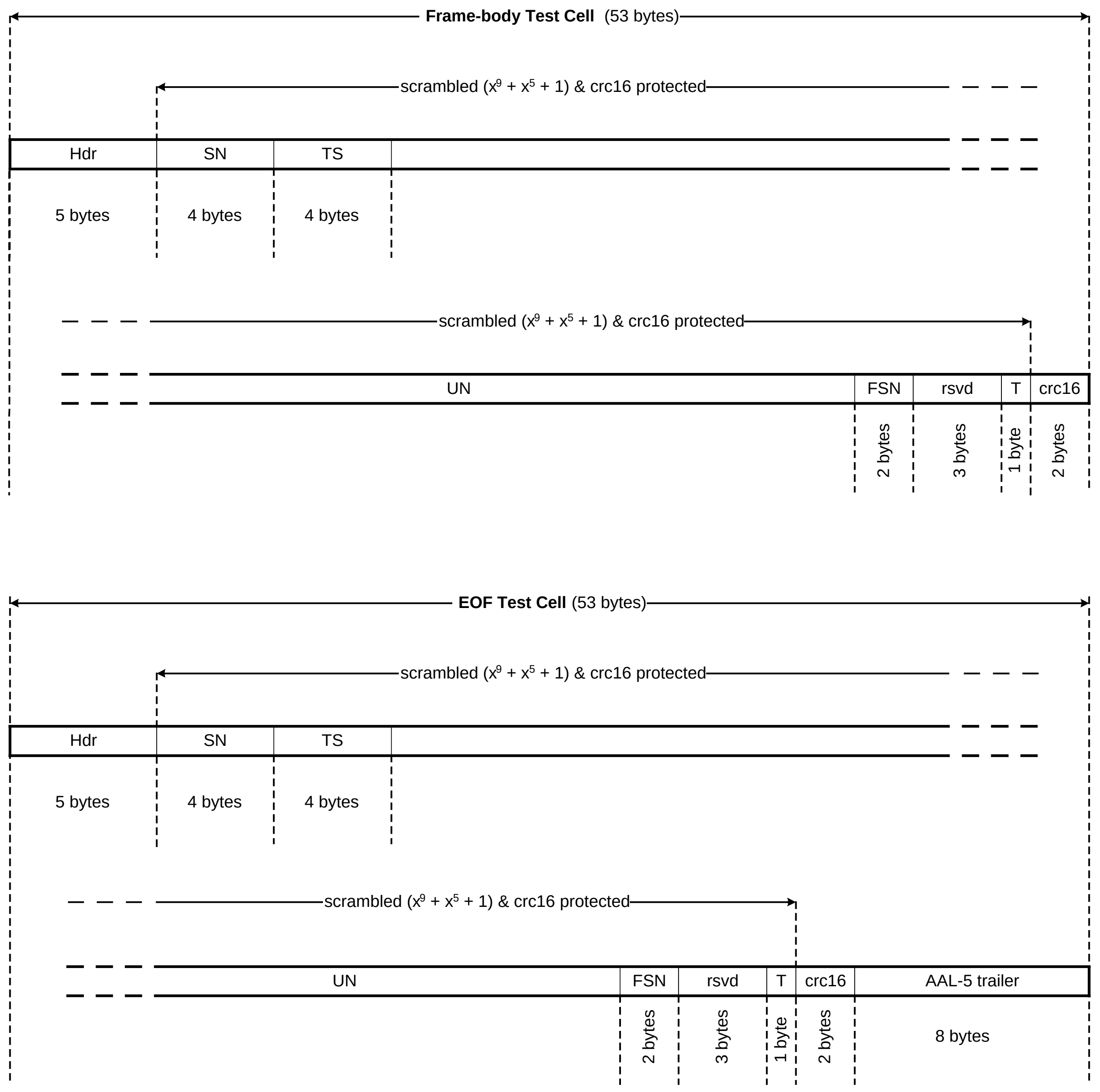
1. all of the test frame features necessary to implement the frame-level performance measurements specified in Section 4.
2. the ability to obtain simultaneous, correlated performance measurements at the ATM and Frame layers.

3. reuse at the frame layer of implementation engines developed for ATM-layer performance measurement.
4. detection of frame payload corruption (via test-cell integrity check) even for SUTs that regenerate the AAL-5 CRC-32 at their egress point.
5. optional, optimized runt frame detection, protecting against the potential miscounting of one errored frame for each runt frame received.

Algorithms that correlate ATM-layer and AAL-5 frame layer measurements are for future study.

D.2.5.1. AAL-5 Test Frame Test Cell Format

The Frame-body test cell is based on that defined in Appendix D.2.2 with the addition of a 16-bit Frame Sequence Number field, long-word aligned at the tail of the UN field, prior and adjacent to the rsvd field.


Frame-body Test Cell (53 bytes)
scrambled (x9 + x5 + 1) & crc16 protected
Hdr
SN
TS
5 bytes
4 bytes
4 bytes
scrambled (x9 + x5 + 1) & crc16 protected
UN
FSN
rsvd
T
crc16
2 bytes
3 bytes
1 byte
2 bytes
EOF Test Cell (53 bytes)
scrambled (x9 + x5 + 1) & crc16 protected
Hdr
SN
TS
5 bytes
4 bytes
4 bytes
scrambled (x9 + x5 + 1) & crc16 protected
UN
FSN
rsvd
T
crc16
AAL-5 trailer
2 bytes
3 bytes
1 byte
2 bytes
8 bytes

Field	Description
Hdr	ATM cell Header
SN	O.191 conformant test cell Sequence Number
TS	O.191 conformant test cell transmit TimeStamp
UN	O.191 conformant UNused octets
FSN	Frame Sequence Number
T	O.191 conformant Test cell payload type (TCPT)
rsvd	reserved
crc16	O.191 conformant test cell CRC-16 payload integrity check

Figure D.5:: AAL-5 Test Frame Format Details

The End-Of-Frame (EOF) test cell has a similar format to the Frame-body test cell except for the last 8 octets. In the EOF test cell, the UN field is eight octets shorter. This leaves room at the tail of the cell for the AAL-5 trailer.

D.2.5.2. Frame Sequence Number

The Frame Sequence Number is a sixteen-bit value that increments by one for each test frame transmitted. The value is encoded as high octet followed by low octet. Two additional octets are reserved to preserve long-word alignment and to allow for future expansion of the FSN.

D.2.5.3. Frame Test Cell Sequence Number

The cell sequence number increments by one for each test cell, is uncorrelated to the FSN, is not reset for each test frame and counts continuously across frame boundaries.

D.2.5.4. Payload Type

The TCTP comprises two fields: PPI and REV. PPI, which is the MSB of TCTP, stands for Proprietary Payload Indicator and must be set to 1 for all test cells in a Test Frame.

D.3. In-service Measurement of QoS Parameters

D.3.1. Introduction

In operational networks, it is necessary for users and network providers to have available in-service monitoring tools to continuously evaluate the performance of the network and services. In-service performance monitoring procedures for ATM networks are specified in ITU-T Recommendation I.610 on OAM principles and functions. OAM performance management cells are inserted and carried in a user cell connection in the forward direction to allow measurement of performance parameters on an end-to-end or segment basis. In the reverse flow, these OAM cells report the monitoring results in the backward direction.

D.3.2. ATM Layer OAM Flows

ITU-T Recommendation I.610 specifies procedures for in-service performance monitoring using F4 OAM flows at the VP-level and F5 OAM flows at the VC-level. OAM cells for the F4 flow have the same VPI value as the user cells of the VPC and are identified by the preassigned values VCI=3 for segment OAM and VCI=4 for end-to-end OAM. OAM cells are generated and inserted at the originating point of the VPC or VPC segment. Intermediate points along the VPC or VPC segment may monitor passing OAM cells and insert new OAM cells, but cannot extract them. F4 OAM cells may be extracted only at the terminating point of the VPC or VPC segment.

OAM cells for the F5 flow have the same VPI/VCI values as the user cells of the VCC and are identified by the preassigned values PTI=100 for segment OAM and PTI=101 for end-to-end OAM. OAM cells are generated and inserted at the originating point of the VCC or VCC segment. Intermediate points along the VCC or VCC segment may monitor passing OAM cells and insert new OAM cells, but cannot extract them. F5 OAM cells may be extracted only at the terminating point of the VCC or VCC segment.

D.3.3. Performance Management Procedures

At the F4 or F5 level, performance monitoring is done by inserting OAM performance management cells at the originating endpoint. In the forward flow, OAM cells carry information about the preceding transmitted user cell block for forward monitoring. Forward monitoring can detect errored cells, lost/misinserted cells, and optionally cell transfer delay. The performance monitoring results are reported in the reverse flow.

At the originating endpoint, a performance monitoring cell insertion request is initiated after every N user cells. The block size N may have the values 128, 256, 512, or 1024. The monitoring cell is inserted at the first available cell location after the request, so the actual size of the monitored cell block may vary from the nominal block size. The cell block size may vary up to a maximum margin of 50% of the value of N for end-to-end performance monitoring, after which a forced insertion becomes necessary. The actual monitoring block size averages out to approximately N cells. Forced insertion for segment performance monitoring is an option.

Performance monitoring can be activated either during connection establishment or at any time after the connection has been established. Activation and deactivation is initiated by the TMN or the end user. After the TMN or end user has requested activation/deactivation, an initialization procedure is needed between the two endpoints of the connection or segment. This initialization procedure serves to coordinate the beginning or end of the transmission and reception of OAM cells, and establish agreement on the block size and direction of transmission to start or stop monitoring. The initialization procedure can be carried out by either (a) using activation/deactivation OAM cells, or (b) entirely via TMN. Further details on activation/deactivation procedures can be found in ITU-T Recommendation I.610. Following performance monitoring activation, the first performance monitoring cell received is used for initialization only and is not used to update performance parameters.

D.3.4. OAM Performance Management Cell

ATM layer OAM cells contain fields common to all types of OAM cells as well as specific fields for each type of OAM cells. Common OAM fields are shown in Figure D.6 and listed below.

- OAM cell type (4 bits): 0010 = performance management;
- OAM function type (4 bits): 0000 = forward monitoring, 0001 = backward reporting;
- function-specific field (45 bytes): described below;
- reserved (6 bits): all zero;
- error detection code (10 bits): CRC-10 computed over the OAM cell payload excluding EDC field.

	cell header	OAM type	function type	function-specific	res	EDC
bits:	40	4	4	45x8	6	10

Figure D.6: Common OAM cell fields

The OAM performance management cell will have the function-specific fields shown in Figure D.7 and listed below.

- Monitoring Cell Sequence Number (MCSN) (1 byte): modulo 256 number to detect the loss of OAM cells;

- total user cell number for CLP=0+1 user cell flow (TUC_{0+1}) (2 bytes): in forward monitoring cell, this is the total number of user cells transmitted prior to insertion of this OAM cell; this number is copied into the TUC_{0+1} field of the corresponding backward reporting OAM cell.
- Block Error Detection Code (BEDC) (2 bytes): even parity BIP-16 code computed over the information fields of block of user cells preceding this OAM cell;
- Total User Cell number for CLP=0 user cell flow (TUC_0) (2 bytes): in forward monitoring cell, this is the total number of CLP=0 user cells transmitted prior to insertion of this OAM cell; this number is copied into the TUC_0 field of the corresponding backward reporting OAM cell.
- timestamp (4 bytes): optional for the insertion time of this OAM cell (default = all 1's); use is for further study;
- unused (29 bytes): all bytes are 0110 1010 (6AH);
- Total Received Cell Count for CLP=0 ($TRCC_0$) (2 bytes): in backward reporting cell, this is total number of CLP=0 user cells received prior to receiving the corresponding forward monitoring cell;
- BLock Error Result (BLER) (1 byte): number of errored parity bits detected by BIP-16 code of the corresponding forward monitoring cell;
- Total Received Cell Count for CLP=0+1 ($TRCC_{0+1}$) (2 bytes): in backward reporting cell, this is total number of user cells received prior to receiving the corresponding forward monitoring cell.

	MCSN	TUC_{0+1}	BEDC	TUC_0	time stamp (opt.)	unused	$TRCC_0$	BLER	$TRCC_{0+1}$
bytes:	1	2	2	2	4	29	2	1	2

Figure D.7: Function-specific fields for OAM performance management cell

D.3.5. Out-of-service Use of Performance Management OAM

In addition to the in-service application, it is proposed that a basic method for out-of-service testing of ATM networks is defined in which end-to-end performance management OAM cells are transmitted and the user data cells are replaced with a dummy load. This dummy load could be a PRBS or fixed pattern. A PRBS would be the best load to use to check network transparency (that is to check that the network carries all bit patterns correctly). In addition to the measurements performed on the performance management OAM cells, a cell or bit error ratio measurement may be performed on the received payload if required.

Performance management OAM capability may be designed into ATM network or end equipment. This is therefore a convenient test method to use if network management systems control the out- of-service test or if test equipment analyzes a test signal generated by a network element.

D.3.6. Benefits of Performance OAM Technique

Use of performance OAM out-of-service would have the following benefits:

- A single test methodology for in-service and out-of-service testing ensures that results obtained in each situation will be comparable.
- Because OAM cells arrive at a lower rate than user cells, there is more time available for processing measurements. This is particularly useful in processor-based measurement architectures.
- Because the sequence number used is eight bits (as opposed to 36 bits in the out-of-service test cell), measurement processing is simpler.
- Because of the widespread deployment of performance OAM techniques in ATM network elements, off-the-shelf devices are readily available to perform these functions.
- Testing an ATM network using performance OAM tests the network's ability to carry the OAM flows as well as user data.
- The backward reporting mechanism would provide a convenient method for test equipment to communicate results.
- Where simultaneous measurements on multiple channels are performed, use of performance OAM allows a mixture of in-service and out-of-service testing to be performed. In this case, channels carrying live traffic can be monitored at the same time as test channels.
- It is a simple test method that will result in economical implementation of low-end ATM test equipment.

Because only the OAM cells carry a timestamp, the cell delay and cell delay variation (CDV) measurements obtained are only of a sampled nature. This should not be a problem when computing mean cell transfer delay, but some extremities of the CDV characteristic may not be seen. By using one-point CDV measurement techniques, the CDV characteristic is obtained for all cells. This is only relevant for a CBR distribution but this is where CDV measurement is mostly required.

D.3.7. ATM Test Cell

Use of performance management OAM out-of-service is a complementary test method to using the ATM test cell. Use of the test cell is necessary to isolate cell loss, misinsertion and mis-sequencing errors more precisely. Also, precise evaluation of cell delay and cell delay variation requires a more frequently transmitted timestamp than that provided by OAM cells.

Appendix E: Examples of Frame Reference Load Models

E.1. Introduction

Appendix E.2 describes (in pseudo-code) how an abstract test case can be generated. Appendix E.3 contains several examples illustrating how parameters might be specified to accommodate a wide range of traffic sources. The representations should be considered as a definitional framework and as such are not related to any specific implementation.

E.2. Pseudo-code Representation of an Abstract Test

The role each configurable parameter plays is illustrated in the following pseudo-code representation of an **abstract** test.

The references to **function_of**(…) mean that the parameter's value is obtained in some manner. The value may be a constant, or may be obtained from some stochastic process, probability distribution, empirical distribution, other representation of a traffic model, *etc*. This is a representation of an **abstract** test, and is intended to illustrate parameter roles and relationships. RLMs need not be structured this way. Examples are provided in the next section.

```
begin test;
 number_phases = function_of(…);
 for I = 1 to number_phases do;      /* generate phases */
  number_active_periods = function_of(…);
  for J = 1 to number_active_periods do;      /* generate active periods */
   number_intervals = function_of(…);
   for K = 1 to number_intervals do;      /* generate characteristic intervals */
    number_frames = function_of(…);
    for L = 1 to number_frames do;      /* generate frames */
     frame_size = function_of(AAL,…);      /* number of cells */
     for M = 1 to  frame_size do;      /* generate cells */
      transfer cell
      if (M <  frame_size) then do; /* no gap after last cell */
       inter_cell_gap = function_of(…);
       wait one inter_cell_gap;
      endif;
     endfor (/*  frame_size */);
     if (L < number_frames) then do; /* no gap after last frame */
      inter_frame_gap = function_of(…);
      wait one inter_frame_gap;
     endif;
    endfor (/* number_frames */);
    if (K < number_intervals) then do; /* no gap after last interval */
     inter_interval_gap = function_of(…);
     wait one inter_interval_gap;
```

```
        **endif;**
      **endfor** (/* number_intervals */);
      **if** (J < *number_active_periods*) **then do**; /* no gap after last active period */
        *inter_activity_gap* = **function_of**(…);
        **wait** one *inter_activity_gap*;
      **endif;**
    **endfor** (/* number_active_periods */);
    **if** (I < *number_phases*) **then do**; /* no gap after last phase */
      *inter_phase_gap* = **function_of**(…);
      **wait** one *inter_phase_gap*;
    **endif**;
  **endfor** (/*  number_phases */);
**end** test;
```

E.3. Examples

Several examples are included to illustrate how parameters might be specified to accommodate various traffic sources.

E.3.1. Simple Single-cell Test

One possible set of parameters is:

number_phases = 1
inter_phase_gap = 0
number_active_periods = 1
inter_activity_gap = 0
number_intervals = 1
inter_interval_gap = 0
number_frames = 1
inter_frame_gap = 0
frame_size = 1 cell
inter_cell_gap = 0

E.3.2. Constant Frame Size, Constant Inter-frame Gap

One possible set of parameters is:

number_phases = 1
inter_phase_gap = 0
number_active_periods = 1
inter_activity_gap = 0
number_intervals = 1
inter_interval_gap = 0
number_frames = <determined by tester>
inter_frame_gap = a constant
frame_size = a constant
inter_cell_gap = <determined by tester>

E.3.3. Constant Frame Size, Variable Inter-frame Gap

One possible set of parameters is similar to Appendix E.3.2, with:

frame_size = a constant
inter_frame_gap = variate from some traffic process, model, distribution, *etc*.

E.3.4. Variable Frame Size, Constant Inter-frame Gap

One possible set of parameters is similar to Appendix E.3.2, with:
frame_size = variate from some traffic process, model, distribution, *etc*.
inter_frame_gap = a constant.

E.3.5. Variable Frame Size, Variable Inter-frame Gap

One possible set of parameters is similar to Appendix E.3.2, with:
frame_size = variate from some traffic process, model, distribution, *etc*.
inter_frame_gap = variate from some traffic process, model, distribution, *etc*.

E.3.6. Persistent (Greedy) Source

One possible set of parameters is:
number_phases = 1
inter_phase_gap = 0
number_active_periods = 1
inter_activity_gap = 0
number_intervals = 1
inter_interval_gap = 0
number_frames = maximum
inter_frame_gap = one *inter_cell_gap* time
frame_size = maximum
inter_cell_gap = computed from allowed rate
cell_transmission_rate = full allowed rate.

E.3.7. Staggered Persistent Source

This is similar to Appendix E.3.6, except that sources start in a staggered fashion. Characteristic intervals can be used to represent the delayed starts. *E.g.*, source 1 starts during interval 1, source 2 starts during interval 2 and joins source 1, source 3 starts during interval 3 and joins sources 1 and 2, *etc*. Phases can also be used to represent the delayed starts. *E.g.*, source 1 starts during test phase 1, source 2 starts during phase 2 and joins source 1, source 3 starts during phase 3 and joins sources 1 and 2, *etc*.

E.3.8. Variable Load Source

This is similar to Appendix E.3.6, except that frames contain one cell. Inter-cell gaps are exponentially distributed.

E.3.9. Bursty Source

Blocks of cells are represented by frames. The frame size is B_{cells}. Inter-frame gaps are exponentially distributed with mean IBI. Bursts and inter-burst gaps can also be modeled by active and idle periods, or by phases.

E.3.10. Three-state Traffic Source

A commonly used traffic source model consists of three states. The source is either active or idle. No traffic is generated while the source is idle. When the source is active, it generates a burst of frames interspersed with short pauses. The burst size is geometrically distributed with mean N_p frames. Frame lengths are geometrically distributed with mean N_b bytes. Pauses between frames are exponentially distributed with mean T_{off}. The lengths of idle periods between active states are exponentially distributed with mean T_{idle}. The frame transmission time is T_{on}.

One possible set of parameters is:

number_phases = 1
inter_phase_gap = 0
number_active_periods = <determined by tester>
inter_activity_gap = exponential(T_{idle})
number_intervals = 1
inter_interval_gap = 0
number_frames = variate ~ geometric(N_p)
inter_frame_gap = variate ~ exponential(T_{off})
frame_size = fixed, or computed from variate ~ geometric(N_b x 8)
inter_cell_gap = <determined by tester>
cell_transmission_rate = derived from T_{on} .

Appendix F: Examples for Reporting Test Results

This appendix provides two tables for reporting test results. The first table refers to all the parameters concerning the configuration, describing the SUT from the HW and SW points of view and also what testing equipment configuration is used during the verification. Diagrams showing the configuration are encouraged. The second table collects instead traffic-related parameters, such as the RLM used for foreground and/or background traffic, the metrics considered and its measured value. Other parameters may be added or a different layout for the table may be used.

Configuration Parameters

Configuration parameters/test cases	Number of ports	Rate of each port	Number of ports per network module	Number of network modules	Number of network modules per fabric	Number of fabrics	SW version	Test equipment	Traffic replication	Inter / Intra module
Test case 1	8	155 Mbps	4	2	2	1	R1	X	Serial	Intra
Test case 2	4	34 Mbps	2	2	2	1	R2	X	Serial	Intra
Test case 3										

Traffic Parameters

Traffic Parameters/ test cases	Type of traffic	Established points	Connection configuration	Service category	RLM parameters	Metric measured	Value	(Optional) Performance enhancing schemes & traffic management functions
Test case 1	Foreground (1 permanent VCC)	Between ports on Different network Modules	n-to-n straight (n=8)	UBR	RLM1 80 Mbps MFL = 100	Throughput (peak)	Y	
	Background (n-1 permanent VCCs , n=8)	Between ports on Different network modules	n-to-n straight (n=8)	CBR	RLM2 40 Mbps MBL = 50	(*)	(*)	

(*) The SUT performance is measured only on traffic carried by the foreground channel, so that metrics and their measured values are not of interest for connections carrying background traffic.

Appendix G: Simple Statistical Methods Recipe

This appendix provides a simple usable recipe for using statistical methods when measuring performance.

G.1. Confidence Interval Estimation

G.1.1. Purpose

By definition, an estimator is not expected to yield the exact value of the parameter that we are trying to estimate (*e.g.,* the mean), but it should usually be equal to this value plus or minus some sampling error. If this sampling error is large, the measured mean may have little value to the performance analyst. For instance assume some measured values for throughput are 2, 2, 2, 18, 18 and 18 frames per second. The mean of this sample is 10 frames per second. Given the large variation in the data, this sample mean may not accurately reflect the true mean. If the test was run longer, it may be the case that all future measurements are 18 frames per second, and therefore the sample mean would approach 18 frames per second in the larger sample.

There are simple statistical methods that estimate how close the measured mean is to the true mean. We can use the estimator's variance and sample size as a measure of this sampling error to help us judge how precise our estimation is. We want an interval of values in which, up to a certain level of confidence, the estimated parameter (mean) is included. This is referred to as a *confidence interval* for our estimated parameter.

G.1.2. Confidence Interval for the Mean

This section explains how to build a confidence interval for a distribution's mean μ. The underlying theory for confidence intervals is beyond the scope of this document. Any number of introductory statistics texts can provide the reasons for this methodology.

G.1.2.1. Conditions

The distribution's mean, μ, and variance, σ^2, exist (*i.e.,* not infinite).

The observations are *i.i.d.* (independent and identically distributed).
If you suspect your observations to be correlated, it is recommended to use one of the several methods that exist to work around this correlation. See, for example, the *batch mean method.*

n is large enough.
In theory, $n > 30$ is often considered large enough. However, the correct value of *n* depends heavily on the original distribution. The more symmetric and Gaussian-like it is, the faster the convergence will be and *n* can be smaller. The more asymmetric, skewed and noisy it is, the slower the convergence will be and the larger *n* must be.

G.1.2.2. Confidence Interval

If our conditions are satisfied, then we have the following asymptotic confidence interval for μ:

$$\mu \text{ is included in } \bar{x} \pm Z_{(1-\alpha)/2}\sqrt{\tfrac{1}{n}\hat{\sigma}^2} \text{ with a confidence level of } \alpha$$

where $Z_{(1-\alpha)/2}$ is the $(1-\alpha)/2$ quantile of the Gaussian distribution with a mean of zero and a variance of one.

G.1.2.3. Walkthrough

1. Collect *n* observations and compute the sample's mean, $\bar{x}$, and variance, $\hat{\sigma}^2$.

$$\bar{x} = \tfrac{1}{n}\sum_{i=1}^{n} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \text{ or, equivalently,}$$

$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i)^2 - \frac{n}{n-1}(\bar{x})^2$$

2. Choose a level of confidence α for your confidence interval and find the corresponding $Z_{(1-\alpha)/2}$. The most common values for a confidence level are 90%, 95%, 99% and 99.9%. The $Z_{(1-\alpha)/2}$ of these values appears in the table of Appendix C.1 (wherein confidence level α is listed under the heading "Confidence" and $Z_{(1-\alpha)/2}$ is listed under the heading "Quantile"). If you want to use another value for α**,** you can find the corresponding $Z_{(1-\alpha)/2}$ in any introductory statistic book and in many statistics-capable software packages.

3. The confidence interval at level α for μ is then

$$\left[\bar{x} - Z_{(1-\alpha)/2}\sqrt{\tfrac{1}{n}\hat{\sigma}^2} \quad , \quad \bar{x} + Z_{(1-\alpha)/2}\sqrt{\tfrac{1}{n}\hat{\sigma}^2}\right]$$

G.1.3. Confidence Interval Using the Bootstrap Method

G.1.3.1. Purpose

The bootstrap method is an alternative method for confidence intervals. It is useful for cases where the estimator's variance is hard (or impossible) to compute, when a sample is too small, or the sample is not well behaved enough to justify using a Gaussian approximation.

The method consists of approximating the real distribution by the collected sample. From this sample we generate many "virtual" samples and compute an estimator for each of them.

G.1.3.2. Conditions

The bootstrap method supposes that the sample's observations are *i.i.d.* (independent and identically distributed). If the observations are correlated, it is recommended to use one of the several methods that exist to work around this correlation. See, for example, the *batch mean method*.

G.1.3.3. Walkthrough

0. Determine θ, the parameter you wish to estimate, and $\hat{\theta}$, its estimator. For instance the sample mean $\bar{x}$.

1. Collect your sample of *n* observations. The choice of *n* is arbitrary, however the larger *n* is, the more precise the approximation of the real distribution.

2. Choose a level of confidence α for your confidence interval. The most common values for a confidence level are 90%, 95%, 99% and 99.9%.

3. From the original sample, draw *n* observations **with replacement**. The probability of drawing any one observation should be 1/n (equally likely). This is commonly referred to as a discrete uniform distribution.
   Note: yes, the n here is the same as in step (1.).

4. Calculate $\hat{\theta}$ for a bootstrapped sample. Keep this value in memory.

5. Repeat (4.) and (5.) *m* times. The number of repetitions *m* should be fairly large (e.g., 1000 times).

6. Sort the obtained estimators $\hat{\theta}_1, \ldots, \hat{\theta}_m$ in ascending order. Let $\hat{\theta}_{(1)}, \ldots, \hat{\theta}_{(m)}$ be the sorted list of estimators.

7. Define
   $$q_1 = \lfloor (1-\alpha)/2 \times m \rfloor + \mathbf{1} \text{ and}$$
   $$q_2 = \lceil (1+\alpha)/2 \times m \rceil$$
   where $\lfloor\ \rfloor$ means round down and $\lceil\ \rceil$ round up.

8. The bootstrapped confidence interval with level α is then
   $$\left[\hat{\theta}_{(q1)} \quad , \quad \hat{\theta}_{(q2)}\right]$$

G.2. Batch Means

G.2.1. Purpose

Many statistical inference procedures require that the observations are *i.i.d.*. Unfortunately, observations coming from most samples are ***not*** independent, but are heavily correlated. The *batch means method* is a simple way to get around correlation and build a set of asymptotically independent observations from correlated (not independent) samples.

G.2.2. Condition

For the batch means method the main condition is that the observations have no long-term dependency. The correlation between two variables should approach 0 as the distance between them increases. In other words, the ability to predict X_{t+k} if we know X_t should decrease as *k* increases. This condition can be violated, for example, if the system under test has a cyclic behavior.

Another condition is to have "enough" observations. "Enough" depends of how well-behaved the original observations are. However, a hundred observations should be deemed a reasonable minimum for most experiments.

G.2.3. Walkthrough

Supposing that the hypothesis of no long-term dependency between the observations is verified, then the method can be applied as follow:

1. Let $x_1, x_2, ..., x_n$ be the original sample.

2. Take *b* and *k*, two integers such as $b \times k = n$. Usually one takes *b* and *k* as balanced as possible (i.e., $b \cong k \cong \sqrt{n}$). The number b is the number of batches, and k is the number of samples in each batch.

3. Divide the sample's x_i into *b* successive batches of *k* observations and computes their mean. That is:

$$\overline{x}_j = \tfrac{1}{k} \sum_{i=(j-1)k+1}^{jk} x_i$$

4. If there is no long-term dependency among the observations and if *k* is big enough, then the batch means $\overline{x}_j$ are *asymptotically independent.*
5. Take the calculated batch means $\overline{x}_j$ as a new sample of size b and calculate a confidence interval using the method described in Appendix G.2.1.

G.2.4. Discussion

It should be mentioned that the batch mean method is a heuristic. That is, its efficiency is not analytically proven. However, it is a method widely used that is known to give good results, as long as the conditions are satisfied and *b* and *k* are large enough.

The method can be justified as follows. Since we assumed that the correlation between successive observations decreases as the distance between the observations increases, the covariance between two successive batch means is mainly composed of the covariance of the observations at the fringes of each batch. By increasing the size of a batch, one reduces the relative importance of these observations. If the batch size is large enough, then the contribution of the fringes will be negligible.

G.2.4.1. How to Choose *b* and *k*?

From the paragraph above, the greater *k* is, the less dependant the batch means will be. However, if *k* is too big, we will have a very small final sample of *b* observations, which can cause a problem if we want to build a confidence interval. The optimal equilibrium between *k* and *b* is different for each experiment, but it is common practice to take *k* and *b* approximately equal.

END OF DOCUMENT