Inho Choi

Ph.D. Candidate
National University of Singapore

Email : inhochoi@comp.nus.edu.sg
Website: https://ihchoi12.github.io/
Mobile : +65-89298595 (Singapore)
+82-1071547231 (Korea)

Education

- **National University of Singapore** — Singapore
  Ph.D. Candidate in School of Computing (Advisor: Prof. Jialin Li); — *Aug 2019 – Aug 2025 (Expected)*

- **Yonsei University** — Seoul, Republic of Korea
  Bachelor's Degree in Computer Science; — *March 2012 – August 2019*

- **Uppsala University** — Uppsala, Sweden
  Exchange Student in Information Technology; — *August 2017 – January 2018*

Publications

- [**ArXiv**] **A Primer on RecoNIC: RDMA-enabled Compute Offloading on SmartNIC**:
  Guanwen Zhong, Aditya Kolekar, Burin Amornpaisannon, **Inho Choi**, Haris Javaid, Mario Baldi.

  ArXiv, Dec, 2023

- [**APSYS '23**] **Capybara: μSecond-scale live TCP migration**:
  Inho Choi, Nimish Wadekar, Raj Joshi, Joshua Fried, Dan R. K. Ports, Irene Zhang, and Jialin Li.

  14th ACM SIGOPS Asia-Pacific Workshop on Systems (APSys 2023)

- [**SIGCOMM '23**] **Network Load Balancing with In-network Reordering Support for RDMA**:
  Cha Hwan Song, Xin Zhe Khooi, Raj Joshi, **Inho Choi**, Jialin Li, and Mun Choon Chan.

  Proceedings of the 2023 ACM SIGCOMM Conference

- [**NSDI '23**] **Hydra: Serialization-Free Network Ordering for Strongly Consistent Distributed Applications**:
  Inho Choi, Ellis Michael, Yunfan Li, Dan Ports, and Jialin Li.

  Proceedings of the 20th USENIX Conference on Network Systems Design and Implementation

- [**S&P (Oakland) '20**] **A Stealthier Partitioning Attack against Bitcoin Peer-to-Peer Network**:
  Muoi Tran, **Inho Choi**, Gi Jun Moon, Viet-Anh Vu, and Min Suk Kang.

  In Proceedings of IEEE Symposium on Security and Privacy, May 2020.

- [**UbiComp Workshop '17**] **Multimodal Data Collection Framework for Mental Stress Monitoring**:
  Saewon Kye, Junhyung Moon, Juneil Lee, **Inho Choi**, Dongmi Cheon, Kyoungwoo Lee.

  In Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers. (Workshop Paper)

Awards

- **NUS Research Scholarship** — Aug 2019 – July 2023
  National University of Singapore

- **Research Achievement Award** — Jan 2023
  National University of Singapore

- **Honors - 1st Semester, 2018** — Aug 2018
  Yonsei University

Experiences

- **Microsoft Research - PhD Research Intern** — Redmond, WA, USA
  Systems Research Group (Mentor: Dan Ports) — *June 2024 — August 2024*
  - **Hardware Accelerator in Data Centers**: Work on a research project of RDMA and computation offloading using FPGA-based SmartNIC

- **AMD - PhD Research Intern** — Singapore
  Xilinx Lab - FPGA / System Design (Mentor: Guanwen Zhong) — *May 2023 - August 2023*
  - **Hardware Accelerator in Data Centers**: Work on a research project of RDMA and computation offloading using FPGA-based SmartNIC

- **National University of Singapore - Research Intern** — Singapore
  Systems & Network Security Lab (Advisor: Min Suk Kang) — *September 2018 - February 2019*
  - **Blockchain Systems & Network Security**: Investigate partitioning attacks against Blockchain network.

- **Yonsei University - Research Intern** — Seoul, Korea
  Dependable Computing Lab (Advisor: Kyoungwoo Lee) — *February 2017 - May 2018*
  - **Sensing Systems using Deep Learning**: Design a realtime stress recognition system using deep learning.

- **Metlife - Summer Intern** — Seoul, Korea
  IT Planning Team — *July 2018 - August 2018*

Hardware-Software Co-Design for High-Performance Data Centers

Many modern applications rely on distributed systems running in large-scale data centers. Distributed systems consist of multiple servers connected in a network whilst ensuring various desired benefits, such as replicating important data or preventing performance bottlenecks by a single server. However, these benefits require properties that are challenging to achieve. For example, distributed systems need to ensure strong data consistency and good load balancing across a large set of servers. Traditional approaches run carefully designed distributed protocols to achieve those properties, but at the cost of heavy network communication among servers.

In recent years, data centers showed a new technological trend where network speeds are constantly increasing while the performance improvement of general purpose processors (CPUs) is slowing down with the end of Moore's law. Consequently, the performance bottleneck of many distributed systems has shifted from the network to CPUs due to the increased network processing burden.

To reduce the overhead on data center servers, two hardware-based approaches are gaining traction in the systems research community recently: (1) programmable network devices (e.g., programmable switches and SmartNICs) for offloading tasks from CPUs to the networks, and (2) kernel-bypassed I/O to reduce the packet processing overhead on CPUs. Although these approaches have demonstrated significant performance benefits, existing solutions involve intricate and problem-specific low-level designs across different hardware components, lacking effective abstractions and often resulting in fragile reliability and correctness.

My research aims to improve the effectiveness and usability of co-designed systems by investigating the interface between specialized network hardware and system software. I have been focusing on three topics that are ubiquitous in data centers. First, distributed consensus protocol for data consistency. Second, load balancing across servers. Lastly, container networking. Putting together, my research envisions high-performance data centers that utilize hardware technologies for strong data consistency and load balancing, while accommodating containerized applications seamlessly. I will describe each topic in more detail below.

Distributed Consensus Protocols

Distributed storage systems scale capacity by partitioning data into multiple shards, and tolerate server failures by replicating each shard. To guarantee strong data consistency, these systems need to ensure a consistent ordering of operations both within each replication group and across shards. Traditionally, consistent ordering is established via layers of distributed agreement protocols, such as Paxos in state machine replication systems and two-phase commit in distributed transactional systems. Each protocol requires multiple rounds of network communication among participants, imposing high performance overheads on the system.

A recent line of work [3, 4] achieves impressive performance improvement in replication and sharded storage systems by co-designing distributed protocols with programmable network devices. Specifically, they propose in-network ordering by offloading the responsibility of request ordering to the network, reducing coordination cost on the host. Unfortunately, prior in-network ordering approaches pose three serious drawbacks due to in-network serialization, i.e., all request traffic must pass through a single sequencer switch. First, in-network serialization negatively impacts network-level load balancing properties. Secondly, the single sequencer can become a scalability bottleneck. Lastly, relying on a single sequencer switch prolongs system downtime during sequencer failures.

To improve the practicality of in-network ordering, one of our recent work, Hydra [1], proposes serialization-free network ordering by using multiple switch sequencers to consistently order requests. Hydra combines per-switch sequence numbers and loosely synchronized physical clocks across switches for message ordering. Hydra receivers establish a consistent order of messages based on stamped physical timestamps, and use stamped sequence numbers to detect dropped messages and ensure progress. Our multi-sequencer design addresses all three drawbacks of prior in-network ordering approaches. Consequently, Hydra improves the average latency of ordered multicast by 13 times through network-level load balancing, scales beyond a single sequencer, and achieves 5x faster sequencer failover.

Load Balancing

Many real-world applications face highly skewed and unpredictable client workloads. For instance, a few celebrities in a social network can attract orders of magnitude more traffic than normal users, and object popularity changes rapidly according to new social trends. These challenging workloads can cause unpredictable load imbalance among servers, leading to both long median and tail service latency. To guarantee strict service level objectives, it is critical for data center systems to dynamically re-balance workloads across servers.

Many prior solutions attempt to address this dynamic load balancing challenge. Pegasus [5] leverages programmable switches to implement selective replication of popular objects, load-aware request forwarding, and dynamic re-replication in case of server overloading. However, Pegasus only functions with UDP transport, and therefore lacks compatibility with TCP-based protocols which are more common in practical deployment. Tackling this limitation, Prism [2] proposes a request-granularity TCP migration system, where a front-end TCP server migrates each request together with the entire connection state to a back-end server to distribute loads. However, Prism not only blocks the TCP connection during migration, but also relies heavily on the Linux kernel APIs, incurring high migration overhead which is prohibitive for many μs-scale data center applications.

In one of our ongoing works, we design a more efficient and versatile live TCP migration system that leverages (1) Demikernel [7], a library OS designed for compatibility with kernel-bypassed I/O devices, and (2) programmable switches. By incorporating our migration protocols entirely into the Demikernel user-level stack, we completely bypass the kernel to minimize migration overhead. Additionally, we co-design programmable switches with our network stack to enable

migration-aware dynamic packet forwarding and temporary packet buffering during migration, further reducing system interference during live TCP migration.

Container Networking

Containers have become an attractive solution to deploy applications in data centers, particularly given the recent popularity of server-less computing. Container networking enables flexible cross-container communication and is available in two modes. In host mode, containers directly use the host network stack and interfaces, which results in sharing the same physical network addresses with the host, imposing management complexities. In contrast, overlay networking mode connects a virtual network of containers through a virtual switch in each host kernel (e.g., Open vSwitch [6]). However, the virtual network's additional per-packet processing results in high performance overheads.

A recent work, Slim [8], removes the per-packet virtual switching overhead by manipulating in-kernel connection metadata during connection setup phase. Despite the reduced per-packet overhead, Slim fails to support connection-less protocols like UDP, or packet-based network policies. Also, Slim significantly increases connection setup time (e.g., 106% longer than a standard overlay networking approach), which is a serious disadvantage for applications with many short connections.

One of my future research directions aims to design a comprehensive and efficient container networking system. Specifically, I propose a new library OS architecture by integrating the container networking layer into the libOS network stack. The layer sits between the application and the network stack transport layer, running completely in user-space. This new integrated design addresses the above issues in existing container networking solutions. First, similar to Slim, the design eliminates per-packet-level virtual network processing overhead. Second, leveraging the transport layer abstraction, it supports all transport protocols including TCP and UDP. Third, it permits per-packet-level network policies as each packet passes through the container networking layer. Lastly, the user-space container network stack design results in highly efficient network I/O operations, including both connection establishment and packet processing.

References

[1] Inho Choi, Ellis Michael, Yunfan Li, Dan R. K. Ports, and Jialin Li. Hydra: Serialization-Free Network Ordering for Strongly Consistent Distributed Applications. In *Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation*, NSDI '23, Boston, MA, USA, 2023. USENIX Association.

[2] Yutaro Hayakawa, Michio Honda, Douglas Santry, and Lars Eggert. Prism: Proxies without the pain. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, pages 535–549. USENIX Association, April 2021.

[3] Jialin Li, Ellis Michael, and Dan R. K. Ports. Eris: Coordination-Free Consistent Transactions Using In-Network Concurrency Control. In *Proceedings of the 26th Symposium on*

Operating Systems Principles, SOSP '17, Shanghai, China, 2017. Association for Computing Machinery.

[4] Jialin Li, Ellis Michael, Naveen Kr. Sharma, Adriana Szekeres, and Dan R. K. Ports. Just Say No to Paxos Overhead: Replacing Consensus with Network Ordering. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI '16, Savannah, GA, USA, 2016. USENIX Association.

[5] Jialin Li, Jacob Nelson, Ellis Michael, Xin Jin, and Dan RK Ports. Pegasus: Tolerating skewed workloads in distributed storage with in-network coherence directories. In *Proceedings of the 14th USENIX Conference on Operating Systems Design and Implementation*, pages 387–406, 2020.

[6] Ben Pfaff, Justin Pettit, Teemu Koponen, Ethan Jackson, Andy Zhou, Jarno Rajahalme, Jesse Gross, Alex Wang, Joe Stringer, Pravin Shelar, et al. The design and implementation of open vswitch. In *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, pages 117–130, 2015.

[7] Irene Zhang, Amanda Raybuck, Pratyush Patel, Kirk Olynyk, Jacob Nelson, Omar S Navarro Leija, Ashlie Martinez, Jing Liu, Anna Kornfeld Simpson, Sujay Jayakar, et al. The demikernel datapath os architecture for microsecond-scale datacenter systems. In *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*, pages 195–211, 2021.

[8] Danyang Zhuo, Kaiyuan Zhang, Yibo Zhu, Hongqiang Harry Liu, Matthew Rockett, Arvind Krishnamurthy, and Thomas Anderson. Slim:{OS} kernel support for a low-overhead container overlay network. In *16th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 19)*, pages 331–344, 2019.

National University of Singapore
Unofficial Transcript

NAME: CHOI INHO **STUDENT NO.:** A0206950H **DATE OF BIRTH:** 27/12/1993 **DATE ISSUED:** 18/01/2023

PROGRAMME: DOCTOR OF PHILOSOPHY
PROGRAMME STATUS: ACTIVE IN PROGRAMME

MODULE		GRADE	CREDITS
ACADEMIC YEAR 2019/2020 SEMESTER 1			
CS5231	SYSTEMS SECURITY	B+	4.00
CS6101	EXPLORATION OF COMPUTER SCIENCE RESEARCH	CS	4.00
ES5001A	GRADUATE ENGLISH COURSE (INTERMEDIATE LEVEL)	B	-
DOCTOR OF PHILOSOPHY			
CUMULATIVE AVERAGE POINT: 4.00			
ACADEMIC YEAR 2019/2020 SEMESTER 2			
CS5321	NETWORK SECURITY	A-	4.00
CS5330	RANDOMIZED ALGORITHMS	A	4.00
CS6230	TOPICS IN INFORMATION SECURITY	A-	4.00
DOCTOR OF PHILOSOPHY			
CUMULATIVE AVERAGE POINT: 4.50			
ACADEMIC YEAR 2020/2021 SEMESTER 1			
CP5010	GRADUATE RESEARCH PAPER	CS	-
CS6203	ADVANCED TOPICS IN DATABASE SYSTEMS	A-	4.00
CS6280	TOPICS IN COMPUTER SCIENCE: SYSTEMS DESIGN FOR NEXT GEN HARDWARE	A+	4.00
DOCTOR OF PHILOSOPHY			
CUMULATIVE AVERAGE POINT: 4.58			
ACADEMIC YEAR 2021/2022 SEMESTER 2			
ES5002	GRADUATE ENGLISH COURSE (ADVANCED LEVEL)	A-	-
DOCTOR OF PHILOSOPHY			
CUMULATIVE AVERAGE POINT: 4.58			

**********END OF TRANSCRIPT**********

Docu. Verification No. ▣ XDC3-0ED0-E27C-F4AD ▣

ACADEMIC TRANSCRIPT

FILE NO. : CME-031468
DATE : JANUARY 16, 2023

NAME IN FULL	: CHOI, INHO	COLLEGE : COLLEGE OF ENGINEERING	
STUDENT NUMBER	: 2012147562	STATUS	: GRADUATED
DATE OF BIRTH	: 1993. 12. 27.	DEGREE REGISTRATION NO	: 연세대2018(학)4568
DATE OF ADMISSION	: MARCH 1, 2012	DATE OF GRADUATION	: AUGUST 30, 2019
DIVISION(SCHOOL)	:		
DEPARTMENT	: DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING		
MAJOR	: COMPUTER SCIENCE AND ENGINEERING		
INTERDISCIPLINARY MAJOR	:		
MINOR	:		
DEGREE	: BACHELOR OF SCIENCE IN ENGINEERING		

TITLE OF COURSE	CREDITS	GRADE
1ST SEMESTER , 2012		
COMPUTER PROGRAMMING*	3	A-
GENERAL CHEMISTRY AND EXPERIMENT(1)	3	B-
CHAPEL(1)	0	P
WRITING	3	B+
COLLEGE ENGLISH I	2	B0
GATEWAY TO COLLEGE	1	P
ATTEMPTED: 12 PASSED: 12 GPA: 3.19		
2ND SEMESTER , 2012		
INTRODUCTION TO ENGINEERING DESIGN*	3	B-
OBJECT-ORIENTED PROGRAMMING	3	B+
CHRISTIANITY & WORLD CULTURE	3	B+
ENGINEERING MATHEMATICS(2)(HONOR CLASS)	3	B+
CHAPEL(2)	0	P
GENERAL CHEMISTRY AND EXPERIMENT(2)	3	A-
WESTERN CIVILIZATION	3	C+
ATTEMPTED: 18 PASSED: 18 GPA: 3.1		
1ST SEMESTER , 2013		
LINEAR ALGEBRA AND ITS APPLICATION	3	A-
DATA STRUCTURES	3	B+
COMPUTER SYSTEM*	3	B+
COLLEGE ENGLISH II	2	B+
ENGINEERING INVESTMENT ANALYSIS	3	C+
CHAPEL(3)	0	P
ATTEMPTED: 14 PASSED: 14 GPA: 3.17		
2ND SEMESTER , 2013		
DISCRETE MATHEMATICS*	3	B+
PROBABILITY AND STATISTICS	3	A0
DIGITAL LOGIC LABORATORY	2	A+
LOGIC CIRCUIT DESIGN*	3	A-
CHAPEL(4)	0	P
ENGINEERING MATHEMATICS(1)	3	A-
MUSIC APPRECIATION	2	B+
ATTEMPTED: 16 PASSED: 16 GPA: 3.71		
WINTER SESSION , 2013		
ENGINEERING ACCOUNTING	3	A0
UNDERSTANDING ON CINEMA	3	B0
ATTEMPTED: 6 PASSED: 6 GPA: 3.5		
1ST SEMESTER , 2014		
OPERATING SYSTEM	3	B0
AUTOMATA AND FORMAL LANGUAGES*	3	B-
COMPUTER GRAPHICS*	3	B-
INTRODUCTION TO CS RESEARCH	1	P
X-design	3	A0
GENERAL PHYSICS AND EXPERIMENT(1)	3	A0
ATTEMPTED: 16 PASSED: 16 GPA: 3.28		
2ND SEMESTER , 2016		
DATABASE MANAGEMENT SYSTEMS	3	A+
ALGORITHM ANALYSIS*	3	A+
READING THE RECOMMENDED BOOKS	1	P
VOLUNTEER SERVICE	1	P
FENCING*	1	P
BIOSYSTEMS & ENVIROMENTAL	3	B+
HUMAN EMOTIONS	3	B+
HUMAN HEALTH	2	B+
ATTEMPTED: 17 PASSED: 17 GPA: 3.73		
TRANSFER CREDITS		
EXCHANGE STUDENT : (공과대 교환) Uppsala University		
1 TERM (09/2017 - 02/2018) TOTAL 14 CREDITS		
1ST SEMESTER , 2018		
ARCHITECTURE OF COMPUTERS	3	A0
SOFTWARE CAPSTONE DESIGN(1)*	3	A0
INTERNET PROGRAMMING	3	A0
VOLUNTEER SERVICE	1	P
PHYSICAL TRAINING*	1	P
ENGLISH COMMUNICATION FOR SCIENG(INT)*	3	A+
GOLF	1	P
ATTEMPTED: 15 PASSED: 15 GPA: 4.08		
1ST SEMESTER , 2019		
LEADERSHIP WORKSHOP	3	P
SOFTWARE CAPSTONE DESIGN(2)	3	A+
INFORMATION SECURITY	3	A+
ENGLISH COMMUNICATION FOR SCIENG(ADV)*	3	A+
BASKETBALL	1	P
ATTEMPTED: 13 PASSED: 13 GPA: 4.3		
TOTAL ATTEMPTED : 127		
TOTAL PASSED : 141		

==== CONTINUE ====

*REMARKS

1. HOURS-PER-WEEK
   ONE HOUR OF CLASS WORK PER WEEK FOR 1 SEMESTER COUNTS FOR 1 CREDIT.
   TWO OR MORE HOURS OF LABORATORY WORK PER WEEK FOR 1 SEMESTER COUNTS FOR 1 CREDIT.
2. WEEKS-PER-YEAR:
   ONE ACADEMIC YEAR CONSISTS OF TWO 16-WEEK SEMESTERS.
3. GRADING SYSTEM:
   BEFORE 1961 : A=100-90, B=89-80, C=79-70, D=69-60, F=59-0
   1961 - 1996 : A=4, B=3, C=2, D=1, F=0
   SINCE 1997 : A+=4.3, A0=4.0, A-=3.7, B+=3.3, B0=3.0, B-=2.7
   C+=2.3, C0=2.0, C-=1.7, D+=1.3, D0=1.0, D-=0.7, F=0
   P: PASS, W: WITHDRAWAL, NP: NON-PASS, WE: WITHDRAWAL FOR EMERGENCY
   S: SATISFACTORY, U: UNSATISFACTORY

SUNG, TAEYOON, Ph.D.
VICE PRESIDENT FOR ACADEMIC AFFAIRS

YONSEI UNIVERSITY 50 Yonsei-ro, Seodaemun-gu, Seoul 03722, Korea Tel (02)2123-2114 https://www.yonsei.ac.kr

Docu. Verification No. ▣ XDC3-0ED0-E27C-F4AD ▣

ACADEMIC TRANSCRIPT

FILE NO. : CME-031468
DATE : JANUARY 16, 2023

NAME IN FULL	: CHOI, INHO	COLLEGE	: COLLEGE OF ENGINEERING
STUDENT NUMBER	: 2012147562	STATUS	: GRADUATED
DATE OF BIRTH	: 1993. 12. 27.	DEGREE REGISTRATION NO	: 연세대2018(학)4568
DATE OF ADMISSION	: MARCH 1, 2012	DATE OF GRADUATION	: AUGUST 30, 2019
DIVISION(SCHOOL)	:		
DEPARTMENT	: DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING		
MAJOR	: COMPUTER SCIENCE AND ENGINEERING		
INTERDISCIPLINARY MAJOR	:		
MINOR	:		
DEGREE	: BACHELOR OF SCIENCE IN ENGINEERING		

TITLE OF COURSE	CREDITS	GRADE	TITLE OF COURSE	CREDITS	GRADE

CUMULATIVE GPA : 3.52 / 4.30
100 POINT SCALE : 91.2 / 100
CLASS RANK (28 / 79)
CUMULATIVE GPA : 3.73 / 4.50

* : English-only class

[HONORS AND AWARDS]
1ST SEMESTER,2018 - HONORS

[English Certification : Advanced]

*** END OF RECORD ***

*REMARKS

1. HOURS-PER-WEEK
   ONE HOUR OF CLASS WORK PER WEEK FOR 1 SEMESTER COUNTS FOR 1 CREDIT.
   TWO OR MORE HOURS OF LABORATORY WORK PER WEEK FOR 1 SEMESTER COUNTS FOR 1 CREDIT.
2. WEEKS-PER-YEAR:
   ONE ACADEMIC YEAR CONSISTS OF TWO 16-WEEK SEMESTERS.
3. GRADING SYSTEM:
   BEFORE 1961 : A=100-90, B=89-80, C=79-70, D=69-60, F=59-0
   1961 - 1996 : A=4, B=3, C=2, D=1, F=0
   SINCE 1997 : A+=4.3, A0=4.0, A-=3.7, B+=3.3, B0=3.0, B-=2.7
   C+=2.3, C0=2.0, C-=1.7, D+=1.3, D0=1.0, D-=0.7, F=0
   P: PASS, W: WITHDRAWAL, NP: NON-PASS, WE: WITHDRAWAL FOR EMERGENCY
   S: SATISFACTORY, U: UNSATISFACTORY

SUNG, TAEYOON, Ph.D.
VICE PRESIDENT FOR ACADEMIC AFFAIRS

YONSEI UNIVERSITY 50 Yonsei-ro, Seodaemun-gu, Seoul 03722, Korea Tel (02)2123-2114 https://www.yonsei.ac.kr

UPPSALA
UNIVERSITET

Official Transcript of Records

Inho Choi
19931227-T399

2018-12-10

Completed courses		Scope	Grade	Date	Note
5PU038	**Intensive Basic Swedish 1, Basic Level A1**	7.5 hp	G	2017-08-28	1
1502	Listening Comprehension and Oral Proficiency	(3.0 hp)	G	2017-08-25	1
1501	Reading Comprehension and Written Profiency	(4.5 hp)	G	2017-08-28	1
1DL360	**Data Mining I**	5.0 hp	5	2017-10-26	2
0200	Assignments	(2.0 hp)	G	2017-10-19	1
0100	Written examination	(3.0 hp)	5	2017-10-26	2
1MD016	**Human-Computer Interaction**	5.0 hp	4	2017-11-07	2
0200	Assignments	(2.0 hp)	G	2017-11-01	1
0100	Group work	(3.0 hp)	G	2017-11-07	1
1DL340	**Artificial Intelligence**	5.0 hp	5	2017-11-20	2
0100	Written examination	(3.0 hp)	5	2017-10-19	2
0200	Assignments	(2.0 hp)	G	2017-11-20	1
1DL321	**Compiler Design I**	5.0 hp	4	2018-01-11	2
0200	Assignments	(1.0 hp)	G	2018-01-11	1
0100	Written Examination	(4.0 hp)	4	2018-01-11	2

60 credits (hp) represent a full academic year.

Notes
1 Grading scale: Pass (G)
2 Grading scale: Pass with distinction (5), Pass with credit (4), Pass (3)

The above is an excerpt from the register of student records.

Harriet Brunnberg
Studentservice

UPPSALA
UNIVERSITET