Analytics and AI on Cray Systems

Session IV

Cray Graph Engine

Session IV

James Maltby, Cray Inc.

Cray Graph Engine (CGE)

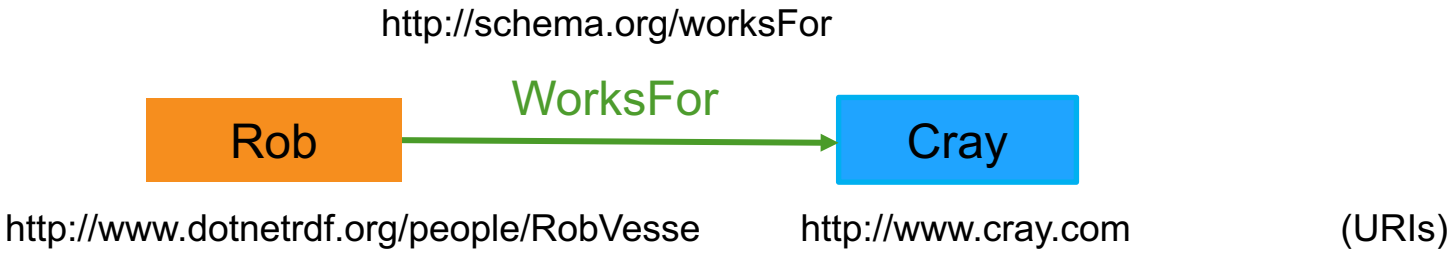
- **Scalable parallel graph analytics framework**
  - Semantic in-memory graph database
    - Basic graph pattern search
    - Graph-theoretic algorithms (whole graph algorithms)
  - W3C Standards Based
    - Uses RDF Data representation
    - Uses SPARQL as query language
  - Built for “vertical scaling” based on parallel and distributed computing principles — competitors are all horizontally scaled
- **Brings interactivity to graph-based discovery**
  - Scaling and performance enables interactive analysis of very large datasets

Cray Graph Engine: Updates and Features

- **Multi-Architecture Support**
  - CGE is available on the Urika-GX and the XC platforms.
  - Strong scaling becomes a key differentiator
    - Bigger datasets => more nodes => better performance
- **Integration with Spark**
  - Interface to data sources - support for end-end analytic workflow realization
- **Integration with Python/Jupyter Notebooks**
  - Connect to SPARQL endpoint using sparqlwrapper or sparql-client packages
  - CGE Python API – utilizes the CGE Java API
    - Start up server, run queries, updates, checkpoint, shut down
- **Integration with R**
  - SPARQL package – connect to SPARQL endpoint, run queries, updates

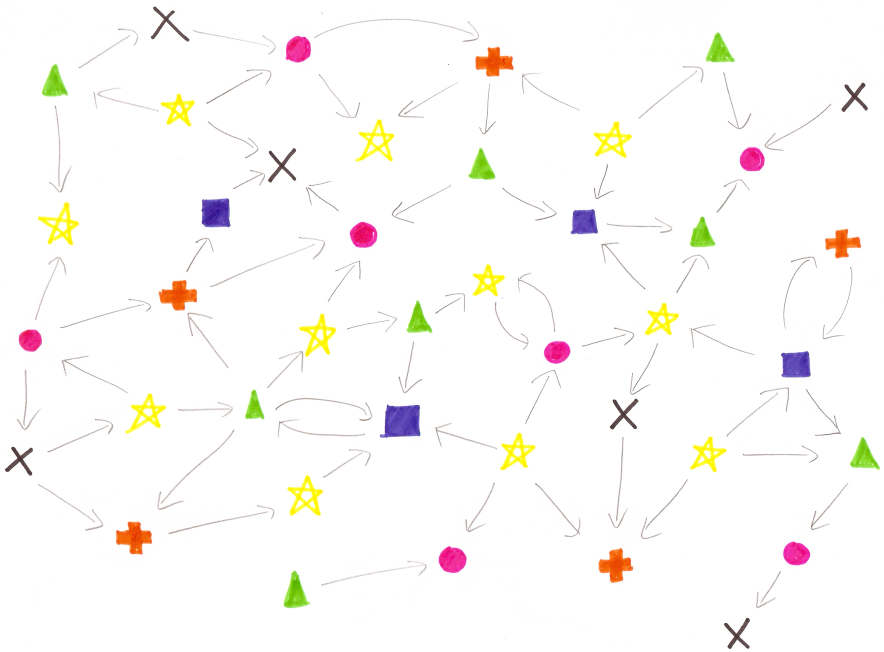
What is RDF?

- **Resource Description Framework (RDF)**
- **A standardized abstract data model centered around the notion of Triples**
- **A Triple expresses a directed relationship between two entities e.g.**



- **Components of a Triple are commonly known as Subject, Predicate and Object**
  - Subject – The thing I am making a statement about
  - Predicate – The relationship being stated
  - Object – The thing which is related

Graph analysis workloads



Two main workloads

- Pattern matching
- Whole graph analysis

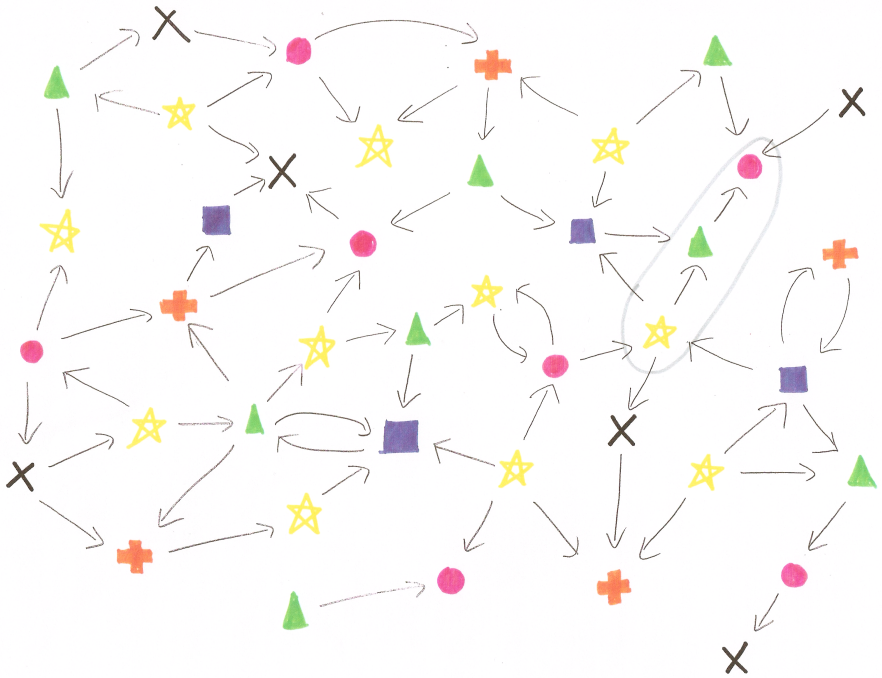
Typical systems only good at one

CGE excels at both

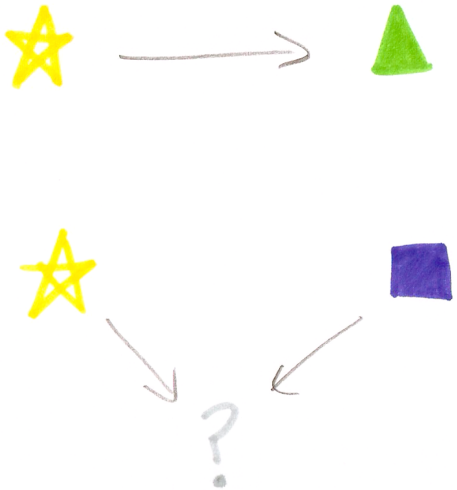
What we plan to cover in the tutorial

- **Background on CGE**
  - Pattern matching, whole-graph analysis
  - Trillion Triples benchmark
- **Hands-on exercises**
  - Build and start up a database (cge-launcher)
  - Run queries
    - Using the cge-cli command line
    - Using the CGE Web UI
  - Integration with R and Python
    - Connecting to the CGE SPARQL endpoint
      - Using R SPARQL package
      - Using Python SPARQLwrapper package

A Graph-pattern matching workload



Given a pattern of interest
find all instances thereof…



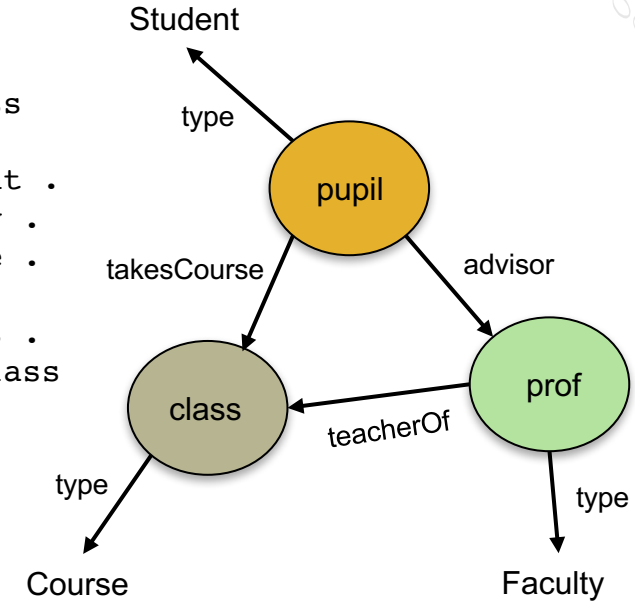
What SPARQL Can Do

Basic Graph Pattern (BGP)

- Primary unit of search for the SPARQL query language
- A SPARQL query always starts with a BGP, followed by additional filters, joins, or other operators
- Subgraph isomorphism problem
- Consists of SCAN, JOIN, MERGE phases

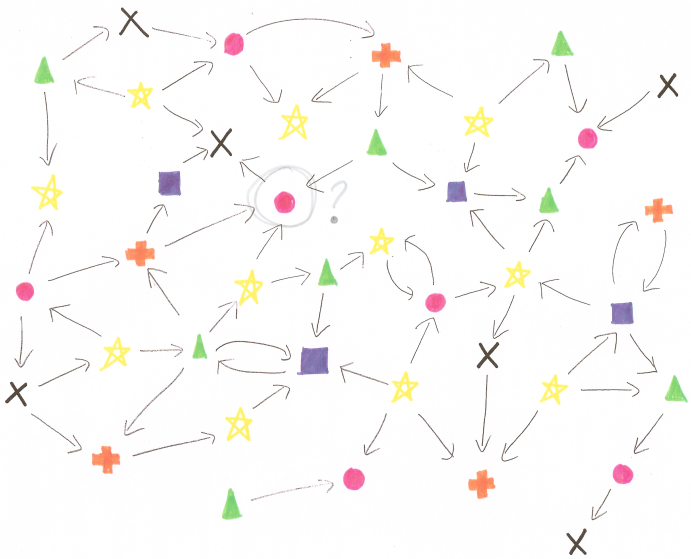
LUBM Query 9

```
SELECT ?pupil, ?prof, ?class
WHERE
{ ?pupil rdf:type ub:Student .
  ?prof rdf:type ub:Faculty .
  ?class rdf:type ub:Course .
  ?pupil ub:advisor ?prof .
  ?prof ub:teacherOf ?class .
  ?pupil ub:takesCourse ?class
}
```

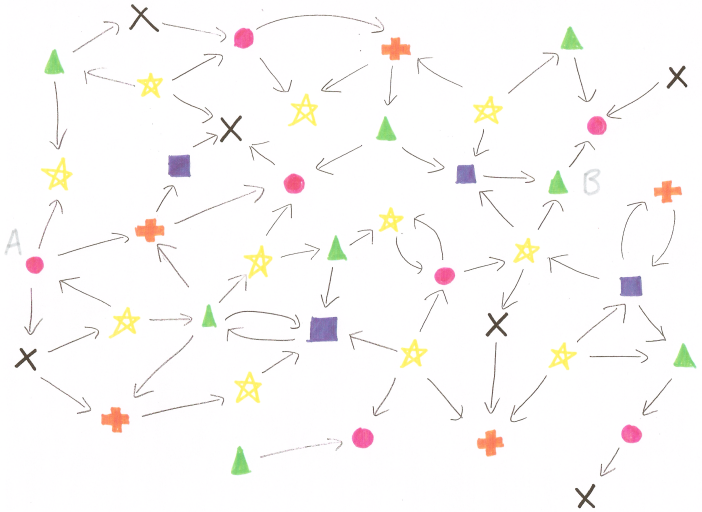


Copyright 2018 Cray Inc.

A Graph-theoretic Workload



What's the shortest route from A to B?



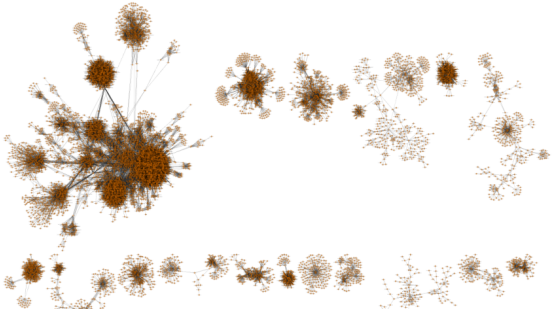
What is the ranking of the targeted vertex?

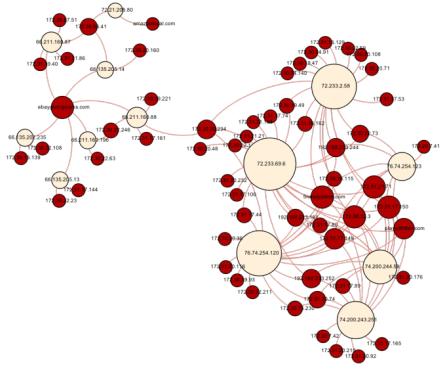
Built-in Graph Functions (BGFs)

- RDF and SPARQL are graph-oriented, but SPARQL is limited in its ability to express graph processing
- We augmented SPARQL with a capability of calling library graph algorithms
- You can go from SPARQL to a graph algorithm and back to SPARQL for further refinement
- The whole is greater than the sum of its parts

Cybersecurity Use Case: Discovering new Cyber threats

- Goal: Proactively identify unknown cyber threats by examining all possible relationships
- Data sets: IP, MAC, BGP, Firewall, DNS, Netflow, Whois, NVD, CIDR…
- Users: Cyber Analysts
- Usage model: Iterative analysis of all patterns across all traffic to explore deviations in frequency of occurrence, derivative patterns of known threats and linking patterns through relationships in offline data

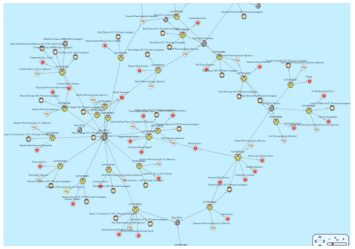


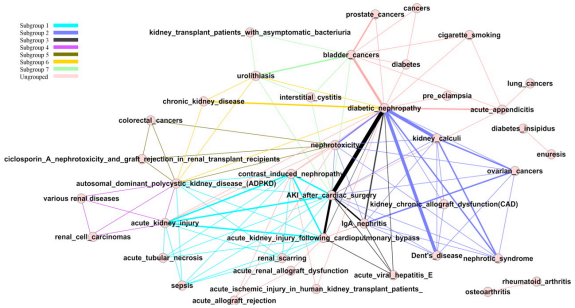


DTG	SIP	DIP	PROT	SPORT	DPORT	PKTS	BYTES	AVG BYTES per PKT	blacklist
2012-07-10T07:52:29	172.31.21.234	212.117.170.54	6	51200	80	6	558	93.00	Tor exit node
2012-07-10T08:37:59	172.31.21.234	212.117.170.54	6	51845	80	6	558	93.00	Tor exit node
2012-07-10T08:38:25	172.31.21.234	212.117.170.54	6	51846	80	8	944	118.00	Tor exit node
2012-07-10T10:59:25	172.31.21.29	212.117.170.54	6	52834	80	7	598	85.43	Tor exit node
2012-07-10T12:04:43	172.31.21.234	212.117.170.54	6	53341	80	6	558	93.00	Tor exit node
2012-07-10T14:04:19	172.31.21.29	212.117.170.54	6	54373	80	6	558	93.00	Tor exit node
2012-07-10T15:59:32	172.31.21.29	212.117.170.54	6	54830	80	6	558	93.00	Tor exit node
2012-07-10T16:45:57	172.31.20.53	65.164.25.47	6	47155	80	33	2347	71.12	SSH Brute Force
2012-07-10T16:45:58	172.31.20.53	65.164.25.47	6	47166	80	8	932	116.50	SSH Brute Force
2012-07-10T16:45:58	172.31.20.53	65.164.25.47	6	47165	80	8	933	116.63	SSH Brute Force

Life Sciences Use Case: Discovering new Drugs

- Goal: Discover the molecular relationships to the disease state
- Data sets: Medline, PubMed, TCGA, Uniprot, Pfam, CRO, Clinical Trials…
- Users: Life sciences researchers both in discovery and development
- Usage model: Identify the complex relationships between the disease state and patients for the development of new biomarkers and drug pathways and use the information for patient stratification for clinical trials, identify new disease targets or companion diagnostics





State of the Art for Trillion Triples

Oracle, September 2014

- First to achieve trillion triples
- Uses a semantic layer over their standard Relational database product
- Disk-based with an SSD cache
- LUBM 4400K, N-Triples format, created 1T triples after inferencing
- Software / hardware:
  - Oracle Database 12c
  - Exadata X4-2 High capacity full rack, 8 DB nodes, 14 storage nodes

Cambridge Semantics, Oct 2016

- Second to achieve a trillion triples
- Dedicated in-memory semantic database, but it is horizontally scaled and more cloud-oriented
- LUBM 4400K, Turtle format, also created 1T triples after inferencing. Data generated and loaded from local SSD on-node
- Software / hardware:
  - Anzo Graph Query Engine (AGQE)
  - 200 nodes Google Compute Platform, each 32 vCPUs, 208 GB

Copyright 2018 Cray Inc.

Comparison of Trillion Triples benchmarks

Software	Load time	Inference time	Query time
Oracle Database 12c	115.2 hours	86.5 hours	22.5 hours
Cambridge Semantics AGQE	1764 seconds	4574 seconds	840 seconds
Cray CGE (2018)	4124 seconds	535 seconds	98 seconds

Copyright 2018 Cray Inc.

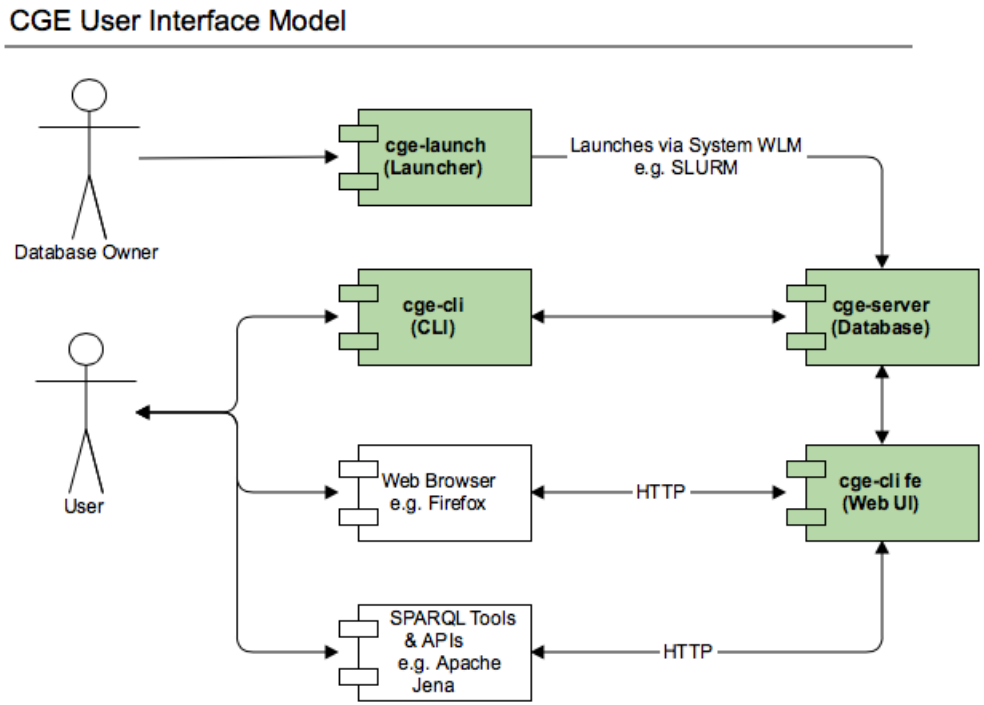
Integrated Life Sciences Dataset

- CGE can accommodate all relevant databases in one environment
- Enables seamless cross-database queries
- Able to load relevant and realistic life sciences datasets in minutes rather than hours or days
  - Build time: 10 minutes (256 nodes)
  - Load time: 105 sec (256 nodes)

Dataset	Size (GB)	# of Triples
Biomodels (r31)	0.2	1057465
Biosamples (v20160912)	61	352661330
ChEMBL (23.0)	75	496019419
Ensembl (Jan 2018)	307	1926736803
Expressionatlas (18-05-2017)	138	685755602
OLS (March 2018)	9.6	73898957
Reactome (r61)	3.6	22354615
UniProt (Feb 2018)	6274	42157935260
OrthoDB (9v2)	137	1128153578
Integrated Total	7 TB	47 Billion

Copyright 2018 Cray Inc.

CGE User Interface Model



- **Database owner launches the database server**
- **Users interact via their preferred interface**
  - Commands Line
  - Web Browser
  - SPARQL Tools & APIs
  - CLI may be used for scripted workflows

Building and launching

- cge-launch is used to build databases:

```
cge-launch –N 8 –I 16 –o /mnt/lustre/myresults –d
/mnt/lustre/mydata –l logfile
```

- `cge-launch` is a script that takes care of resource allocation for the user!

- After a successful build, the database directory will contain:

dataset.nt
rules.txt
dbQuads
string_table_chars
string_table_chars.index
graph.info

The database port

- A TCP port used for communication with this server instance:

```
cge-launch –N 8 –I 16 –p 3750 …
```

- The default is 3750
- Changing this port allows multiple versions

The database directory

- The database directory, typically:

```
/mnt/lustre/user/datasets/lubm0
```

- Is the start of a directory tree containing all checkpoints, and potentially authorized_keys
- It can be moved, archived and returned (!)
- Multiple users can access it, with permissions

The Command Line Interface (CLI)

```
cge-cli –db-port 3750 query myquery.rq
```

- The CLI is used for most interactions with the server, and has many options…
- `cge-cli help` (or `cge-cli help checkpoint`) will give verbose information on options
- Designed for scripted control, querying and updates with database server
- Communications are secure SSH

CLI — most common options

`query` – submits SPARQL queries

`update` – submits SPARUL updates

`sparql` – submits both queries and updates

`checkpoint` – creates a database checkpoint

`echo` – check status of server

Customization using NVPs and cge.properties file

- **Retrieve the Default NVP Configurations**

```
$ cge-cli nvp-info
```

- **For Piz Daint, need to modify internal memory allocator defaults settings due to accommodate smaller 64GB nodes**
  - CGE uses a internal memory allocator to avoid issues with observed memory fragmentation on XC systems
  - cge.server.BuddyMemPercent 20 (current default 35)
  - cge. server.PersistBuddyMemPercent 20 (current default 25)

- **More information**
  - https://pubs.cray.com/content/S-3014/3.2.UP01/cray-graph-engine-user-guide

Hands on Exercises: Running CGE on Piz Daint

- **See README for instructions and exercises**
  - /scratch/snx3000/kristyn/CUG2018/CGE/README
- **Load CGE module**
  - module use /scratch/snx3000/kristyn/CUG2018/modulefiles
  - module load cge
- **To use CGE Web UI, need to set up ssh tunneling**
  - Piz Daint is only accessible via ssh from the front end ela.cscs.ch,need to create a tunnel through the front end to Daint
    - Use a random port number (8022) to connect to ssh port 22
      - ssh -L localhost:8022:daint:22 ela.cscs.ch
    - Then ssh directly into daint node, choosing another random port number (15000) for CGE fe
      - ssh –p 8022 –L localhost:15000:localhost:15000 localhost

Hands on Exercises: Running CGE on Piz Daint (2)

- **Create cge.properties file in ~/.cge/cge.properties**
- **Set up database directory on Lustre**
  - Make sure Lustre striping is set
    - lfs setstripe –c 16 --stripe-size 16m .
  - Needed files: dataset.nt, graph.info, rules.txt
- **Set up query_results directory on Lustre**
  - Make sure Lustre striping is set
- **Be sure to set passwordless ssh**
  - ssh-keygen
  - cat id_rsa.pub >> authorized_keys

Q & A

James Maltby, Cray Inc.
jmaltby@cray.com

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publicly announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, URIKA and YARCDATA. The following are trademarks of Cray Inc.: CHAPEL, CLUSTER CONNECT, CLUSTERSTOR, CRAYDOC, CRAYPAT, CRAYPORT, DATAWARP, ECOPHLEX, LIBSCI, NODEKARE, REVEAL. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used on this website are the property of their respective owners.

Q & A

Overall Session