Research –
Personal Stories and Specifics

Prof. Dorian Arnold
University of New Mexico
The (hopefully not too bland) Story of Dr. Dorian
It All Started in Belize (sort of)
Regis University (aka How I got into CS)

- Denver, CO
- Catholic Jesuit
- ~1700 undergrads

- How the hell ...?
  “Excuse me, father ...”

- Why CS?
How I chose UTK (or vice versa)

- What not to do when selecting a graduate program
  - Don’t stay on an island as you ponder
  - Don’t be ignorant of what graduate school is all about
  - Don’t focus on the short term costs
  - Don’t be overly concerned with geography
  - Fallbacks may be “less safe” ... for funding anyway

- Although ... sometimes it works out anyway
V-O-L-S: Go Vols Go!

- Took “real” systems classes for the first time.
  - And liked them 😊

- Research asst. for my fav. prof. (“Dr.” Jim Plank)
  - And liked it

- And the Vols were cool!
Checkpoint/restart
- Mechanism for tolerating process/task failures
- Periodically save checkpoint to stable storage
- If task fails, resume it from most recent checkpoint

How can we **improve checkpoint/restart efficiency**?
- Smart file formats
- Leverage copy-on-write for **asynchronous checkpointing**

Also, my first cool acronym
- **CLUBS**: Checkpoint Library for Unix-Based Systems
How Dolly Parton influenced my Career
A 2.5 Year Post-Masters Research Stint

Why?

◦ I was learning the game
◦ I now knew the good schools (at least by reputation)
◦ I knew I wanted to be in academia
◦ I knew academics cared about “pedigree”

◦ Oh yeah … and I had the chance to work with one of the most famous computer scientists in the world.
Research @ Innovative Computing Lab

- Directed by Prof. Jack Dongarra
- NetSolve project
  - Fault-tolerant, RPC system for scientific computations
- My contributions:
  - Optimized client/server dataflows
  - Logistical computing using distributed storage technologies
  - Improved scheduling using network performance predictions
- NetSolve won an R&D 100 award in 1999!
How I chose Wisconsin (Really!)

- Post masters research strategy worked.
  - I applied to 10 schools (all top 25)
  - I got accepted to 7.5
  - Wisconsin had the best weather 😊
Research @ Wisconsin

- Scalable Middleware Infrastructure
- Fault-tolerant Computing
- Scalable Debugging
**MRNet: Multicast/Reduction Network**  
Scalable Communication and Data Aggregation

**MRNet Overview**
- Tree-Based Overlay Network for scalable applications
- Process hierarchy for efficient group communication
- In-network aggregation for efficient data analysis
- Fully customizable topologies and filters
- Open-source software
- C++ API

**Sample MRNet Projects**
- TBON-FS (University of Wisconsin)
- Stack Trace Analysis Tool (LLNL)
- TauOverMRNet (University of Oregon)
- Open|SpeedShop (Krell Institute)
- CEPBA-Tools (Universitat Politècnica de Catalunya)
**Robust, Scalable Data Aggregation**

*State Compensation* uses redundant information below failure zones to compensate for lost computational and communication state.

---

**Normal Operation**
- No explicit state replication
- Filter state captures input history
- Send filter state to new parent

**Failure Detection**
- Child fails? Notify non-failed peers
- Parent fails? Orphaned sub-trees reconnect

**Tree Reconstruction**
- Orphans select new parent without coordination
STAT is a lightweight debugging aid that uses stack traces to classify process equivalence and profile application.

Thousands of tasks reduce to few classes.

Analyze representatives with full debugger

Temporal analysis determines tasks’ relative progress
UNM: The Final? Frontier

- Co-director of the Scalable Systems Lab

- General research in large scale distributed systems:
  - Continuing all the previously mentioned research directions
  - And, of course, adding some new ones ...

- Fundamentally grounded solutions to real problems!
  - Generate useful software artifacts whenever plausible.
Lightweight Infrastructure-Bootstrapping Infrastructure

Given a node allocation, efficiently **start** an infrastructure’s composite processes and **propagate** necessary initialization information.

**LIBI** is a **generic service for scalable process launch and information dissemination**.
Scalable Checkpoint/Restart

As HPC systems get larger and more complex, system mean time between failure decreases.

Checkpoint/restart, the most common HPC fault-tolerance mechanism, does not scale in current form.

We are studying checkpoint/restart enhancements for more scalable:

- Checkpoint compression
- Checkpoint aggregation
- Checkpoint/restart + task replication
- Hash-based incremental checkpointing
- ...

\[\text{Checkpoint Compression Viability} \quad \text{compression} \_\text{factor} \times \text{compression} \_\text{speed} < \text{commit} \_\text{rate}\]
**Autonomous Middleware**

**Goal**
Efficient, scalable systems from:

- System Knowledge in Application
- Application Knowledge in System

**Approach: Dynamic, Autonomous Operation**
Self-configuring: Automatic TBÖN topology configuration

- Self-monitoring: TBÖN health and performance
- Self-healing: TBÖN Fault tolerance and failure recovery
- Self-optimizing: Dynamic TBÖN reconfiguration to improve performance

**Challenges**
- Reliable service at scale
- Choosing the “best” TBÖN topologies?
  - Load and system characteristics may vary over time
- Online improvement of TBÖN performance?
  - Throughput, latency, resource consumption, startup costs, ...
- Flexible, elegant solution space

**Monitoring**
- Symptoms
- Detecting
- Deciding
- Decisions
- Monitoring
- Sensing
- Effectors
- Acting
- Actions
Questions

- "Were there moments where you weren't sure if you wanted to pursue a PhD? If so, what changed, how did you know you wanted to pursue a PhD?"

- Did you have to find a mentor aside from your PI in order to gain better feedback and support throughout your time as a graduate student?

- What advice do you have for first year graduate students in terms of settling into a research lab?
Feel free to contact me (darnold) (cs.unm.edu)

Apply to UNM! 😊
  ◦ New Mexico is great!
  ◦ The department is great!
  ◦ Why not!