DXC'11 – Third International Diagnostic Competition

Scott Poll, Adam Sweet - NASA Ames Research Center

Sriram Narasimhan, Matt Daigle - University of California, Santa Cruz @NASA Ames

Johan de Kleer, David Garcia, Tolga Kurtoglu – PARC

Alexander Feldman – Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Alberto Gonzalez-Sanchez – Delft University of Technology

Rui Abreu – University of Porto, Portugal

Agenda

DXC Competition

Participant Talks

- AntigenDX (Jeremy Mange?)
- QED/QED-PC (Matt Daigle)
- NGDE* (Johan de Kleer)
- Ochai (Rui Abreu)
- Raptor* (Alberto Gonzalez-Sanchez)

Q & A, Future of DXC

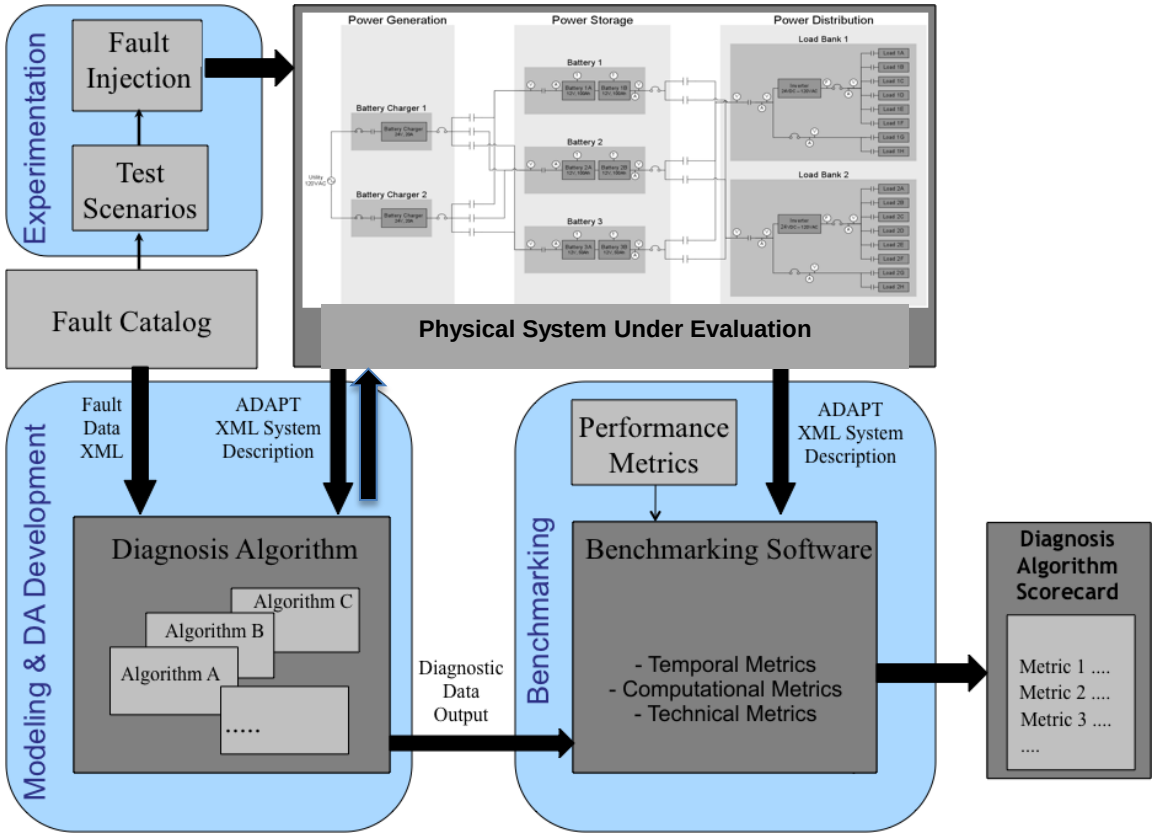
DXC'11 Objectives

- Compare DAs on a standardized platform
- Facilitate maturation of diagnostic technologies
- Mimic use of DAs in operational context
- Identify open issues in the DXC framework
- Note observations to support future competitions

DXC'11 Setup

- Diagnostic competition based on DXC Framework
- Diagnosis Algorithms (DAs) competing on common base
- 4 diagnostic problems announced, motivated by use case
- New software track this year
- DAs in 3 of 4 diagnostic problems (DP-I, DP-III, DP-IV)
- Quantitative evaluation to rank DAs

Framework

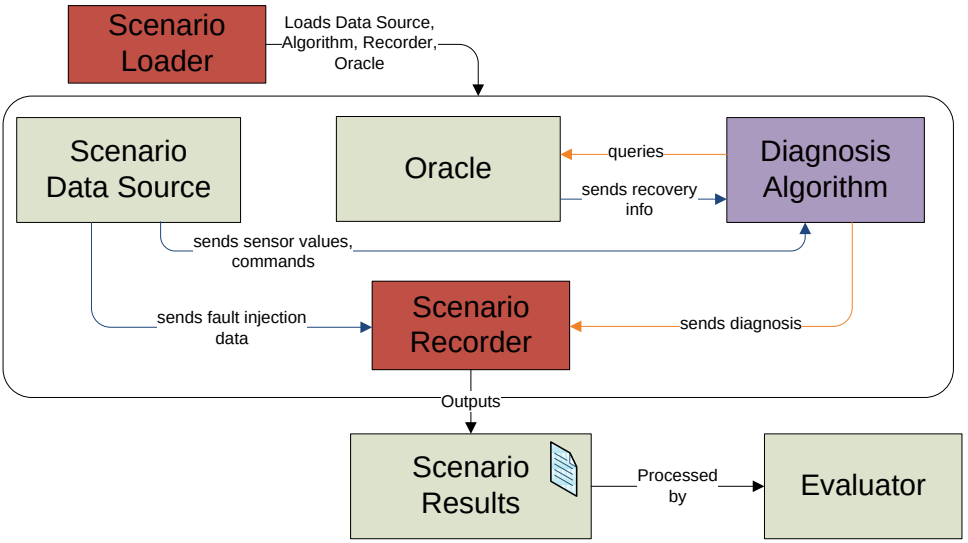


Standard formats for system description, data, & diagnosis results

Run-time architecture to execute DAs and compute metrics

Performance metric definitions

DXF Run-time Architecture



API for communication between DA, Scenario Data Source, and Scenario Recorder

Recorder records results from DA, key time points, & OS metrics (memory & CPU time)

Evaluator implements computation of some common metrics

Framework Usage for Competition

Generate system description & sample data

System description and sample data for ADAPT, ISCAS85 circuits, software programs

↓

Develop/Update diagnosis algorithm(s)

Create DAs (build models, rule base etc.)

↓

Write wrapper code to get data and publish diagnosis results

↓

Generate Scenarios

↓

Execute DA(s) in evaluation framework to compute metrics

↓

Analyze metrics & compute comparative performance assessment

Evaluate and score DAs

Systems

Track	DP	Systems	Operational Scenario	Diagnostic Use Case
Industrial	I	ADAPT-Lite	Single-string UAS Mission	Abort Recommendation
	II	ADAPT	Redundant Systems UAS Mission	Fault Recovery Recommendation
Synthetic	III	ISCAS85	Digital System Maintenance	Trouble Shooting Support
Software	IV	Siemens, SIR	Software Verification	Software regression testing

Diagnosis Algorithms (DAs)

DA	DP	Algorithm Type
AntigenDX	I	AIS+rule-based
QED	I	Model-based, global
QED-PC	I	Model-based, local
NGDE-NP	III	Model-based
NGDE-E	III	Model-based
NGDE-F	III	Model-based
Optimist	III	Edge-case DA
Probee	III	Edge-case DA
Ochiai	IV	Similarity coefficient
Raptor-H	IV	Model-based + Bayes
Raptor-EPS2	IV	Model-based + Bayes

Industrial Track

Scott Poll

Matt Daigle

David Garcia

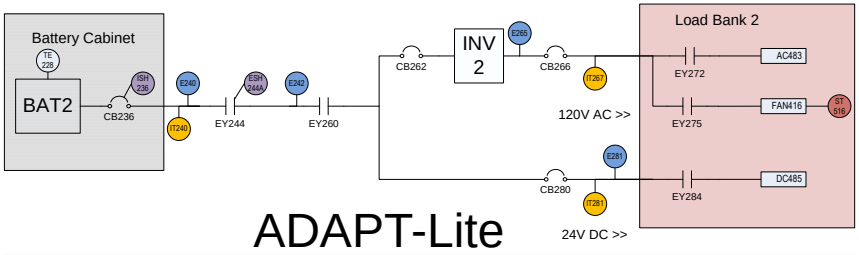
Tolga Kurtoglu

Sriram Narasimhan

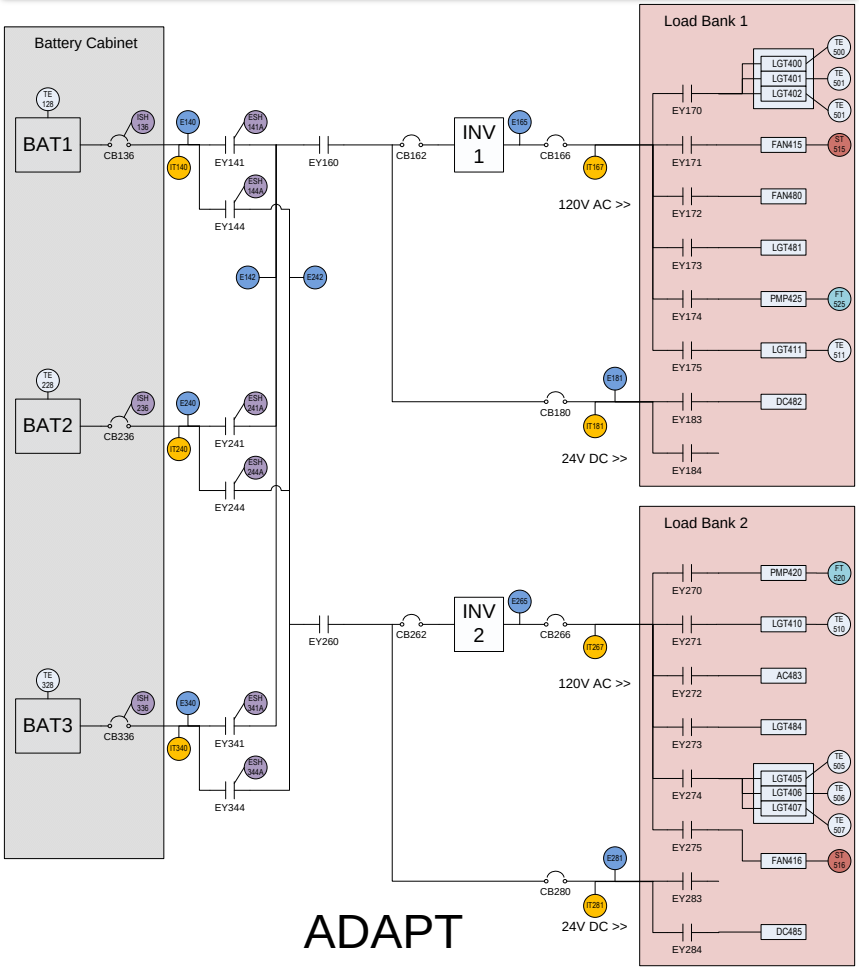
Adam Sweet

Industrial Track – ADAPT EPS

Aspect		DP-I	DP-II
System		ADAPT-Lite	ADAPT
Operational scenario		Single-string UAS mission	Redundant systems UAS mission
Diagnostic use case		Abort rec.	Fault recovery rec.
Number of components		25	96
Number of modes		102	306
Initial relay state		Closed	Open
Initial circuit breaker state		Closed	Closed
Nominal mode changes		No	Yes
Multiple faults		No	Yes
Fault types	Discrete	Yes	Yes
	Offset	Yes	Yes
	Drift (incipient)	Yes	No
	Intermittent offset	Yes	No

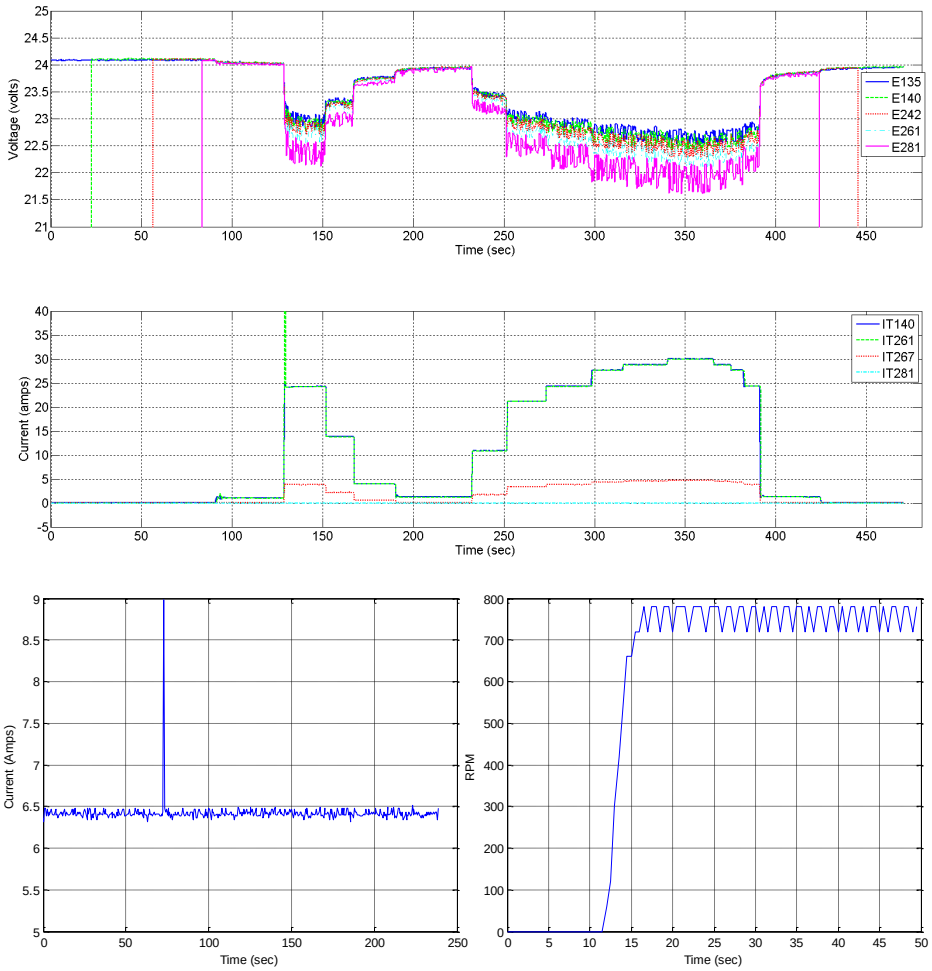


ADAPT-Lite



ADAPT

Industrial Track Diagnosis Challenges



- Multiple modes of operation (DP-II)
- Continuous dynamics
- Data transients (DP-II)
- Load-dependent sensor noise
- Sensor quantization

DXC'11: Rematch of DXC'10

Reduced Sensor set

- ADAPT-Lite, 20 (DXC'09) -> 11
- ADAPT, 83 -> 40

Multiple sample rates 1, 2, 10 Hz

- DXC'09 had all data at 2 Hz

Decision support strategy

- Deal with ambiguities

New fault types

- Drift (incipient)
- Abrupt intermittent

Industrial Track Scenario Generation

#Scenarios	DP-I
Nominal	30
Single-fault	197
Abort	96
No-abort	131

Each scenario ~4 minutes of experimental data (nominal or faulty) collected at 10 Hz (subsampled to 1, 2 Hz)

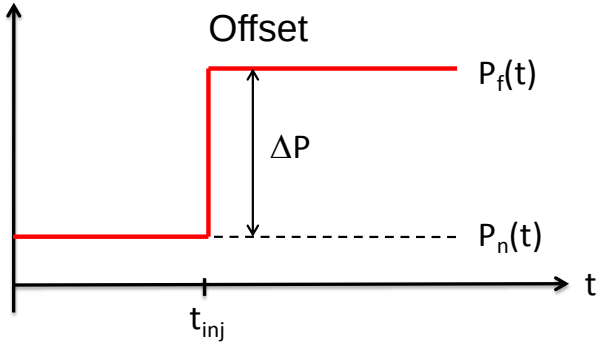
Initial operating state specified

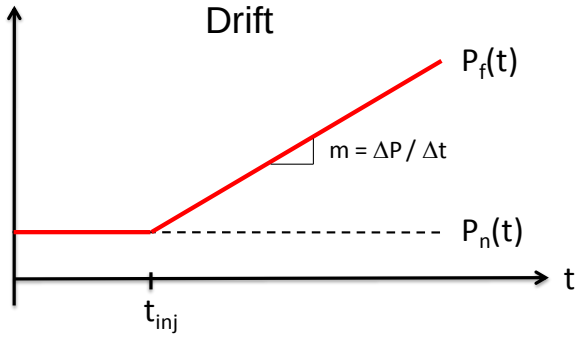
No faults injected in first or last 30 seconds to allow for DA start-up and shut-down

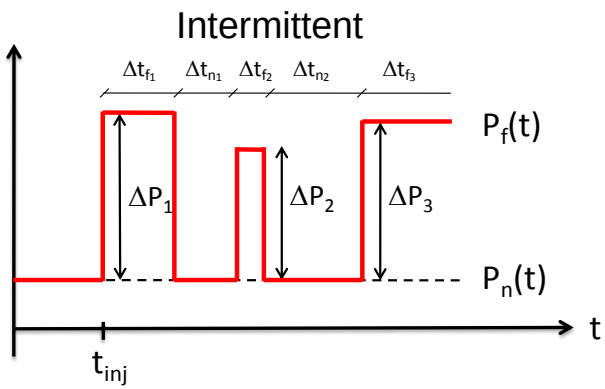
DP-I Fault Profiles and Abort Conditions

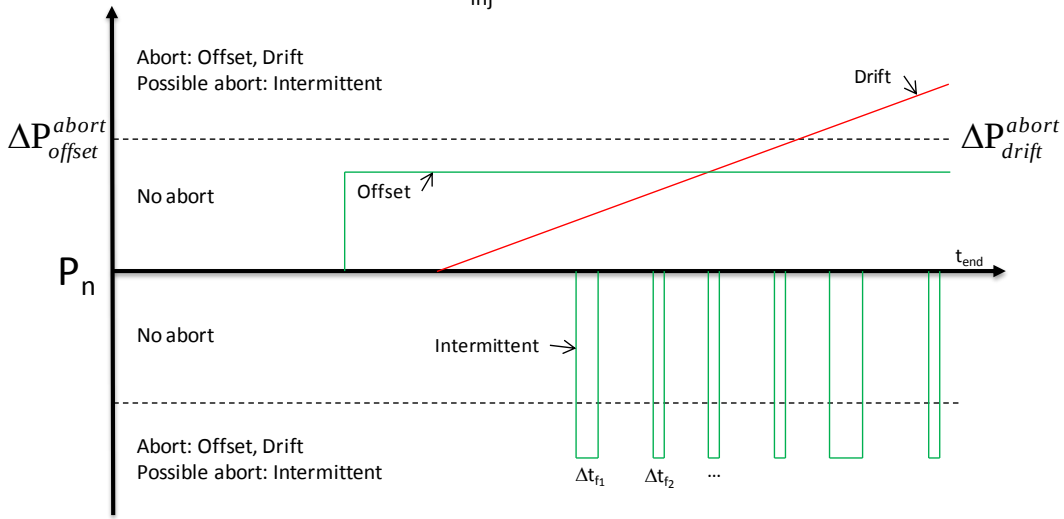
Any fault which cuts off power to loads => abort

Other faults are conditional on fault magnitude:









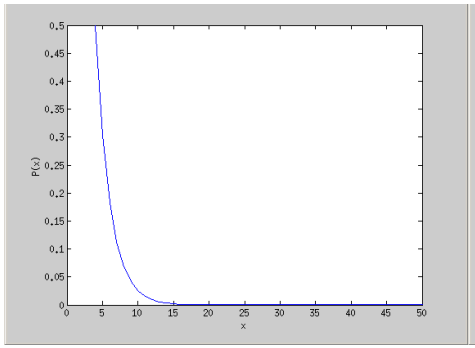
3 examples illustrated

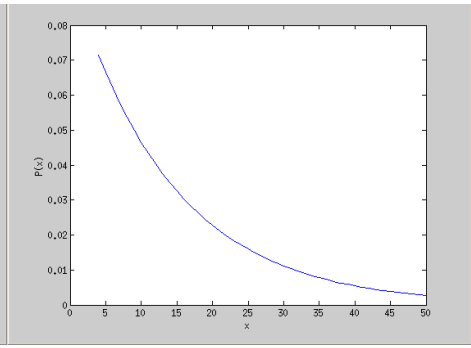
Intermittent Faults

- Simplified intermittent fault behavior
  - Assume constant offset from nominal value during a given fault occurrence (but not constant between occurrences)
  - Assume that the minimum fault duration and fault wait time is not less than 2 seconds
  - Assume severity of intermittent fault is characterized by magnitude and the percentage of time that the behavior is faulty
- Use shifted exponential distributions for fault duration and fault wait time

$$f(x;\theta,\sigma) = \frac{1}{\sigma} e^{-(x-\theta)/\sigma}; x \geq \theta, \sigma > 0$$

θ : minimum x; $\theta + \sigma$: mean x





DP-I Faults

Type	Subtype	Fault	#	abort / no-abort
nominal		no fault	30	0 / 30
circ. breaker		failed open	6	6/ 0
inverter		failed off	2	2 / 0
load	fan	failed off	2	2 / 0
		over speed	2	2 / 0
		under speed	1	0 / 1
	AC load	resistance offset	11	5 / 6
		resistance drift	12	6 / 6
		intermittent res. offset	11	5 / 6
		failed off	1	1 / 0
	DC load	resistance offset	12	6 / 6
		resistance drift	13	6 / 7
		intermittent res. offset	12	6 / 6
		failed off	2	2 / 0
relay		stuck open	7	7/ 0
sensor	position	stuck	3	0 / 3
	current, temp., voltage	offset	31	13 / 18
		stuck	11	6 / 5
		drift	26	10 / 16
		intermittent offset	32	11 / 21
		Totals:	227	96 / 131

227 scenarios

- 96 abort / 131 no-abort

Single faults

Fault Injection types

- Software induced faults
  - Inject spurious commands to relays to cause unintended mode changes
  - Apply transformations to sensor values (offset, stuck, etc.)
- Hardware induced faults
  - Physical injection at the testbed hardware (trip circuit breaker, turn off inverter, etc.)
  - Programmable electronic load for parametric load faults

Industrial Track Scoring

DP-I, Decision Cost

Actual Case / DA rec.	abort	non-abort
abort	0	$c_{mission}$
non-abort	$c_{mission} + c_{vehicle}$	0

$$c_{mission} = 25$$
$$c_{vehicle} = 100$$

Decision Costs

- Losing the mission
- Losing the mission and vehicle

Sum costs over all scenarios

DA with lowest total cost wins

Industrial Track Results

DP-I, DXC'11

DA	Cost	Rank
AntigenDX	350	2
QED	50	1
QED-PC	375	3

For comparison,
DA that always aborts 131 * 25 = 3275,
DA that never aborts 96 * 125 = 12000,
Minimum possible cost = 0

DP-I, DXC'10

DA	Cost	Rank
AdaptedFACT	2250	2
HyDE-A	6950	6
ProADAPT	4925	5
QED	2350	3
SystemicsC	2400	4
TARDEC	2000	1

For comparison,
DA that always aborts 89 * 25 = 2225,
DA that never aborts 65 * 125 = 8125,
Minimum possible cost = 0

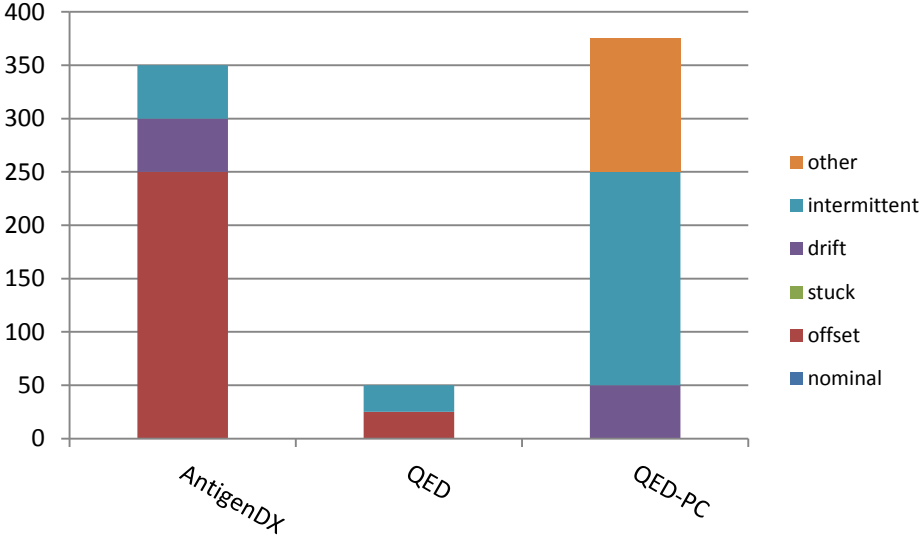
DP-I, DXC'09 Metrics

DP-I

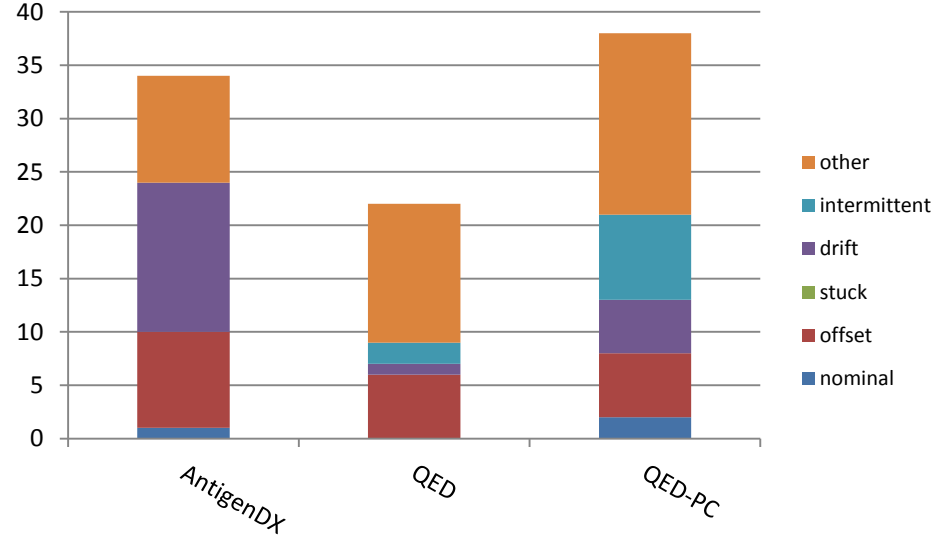
DA	M_{fd} (s)	M_{fn}	M_{fp}	M_{da}	M_{fi} (s)	M_{err}	M_{cpu}(ms)	M_{mem}(kb)
AntigenDX	62.8	0.005	0.009	0.987	62.8	34	680	1935
QED	10.5	0.010	0.004	0.987	124.1	22	824	5356
QED-PC	15.2	0.020	0.026	0.956	123.9	38	589	5320

Metric	Name	Class
M_{fd}	fault detection time	detection
M_{fn}	false negative scenario	detection
M_{fp}	false positive scenario	detection
M_{da}	scenario detection accuracy	detection
M_{fi}	fault isolation time	isolation
M_{err}	classification errors	isolation
M_{cpu}	CPU load	computation
M_{mem}	memory load	computation

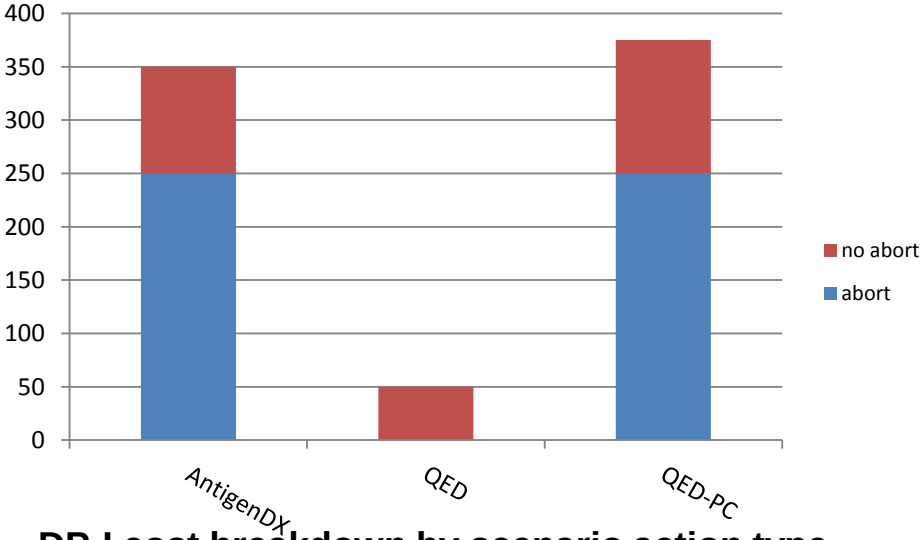
Results – DP-I



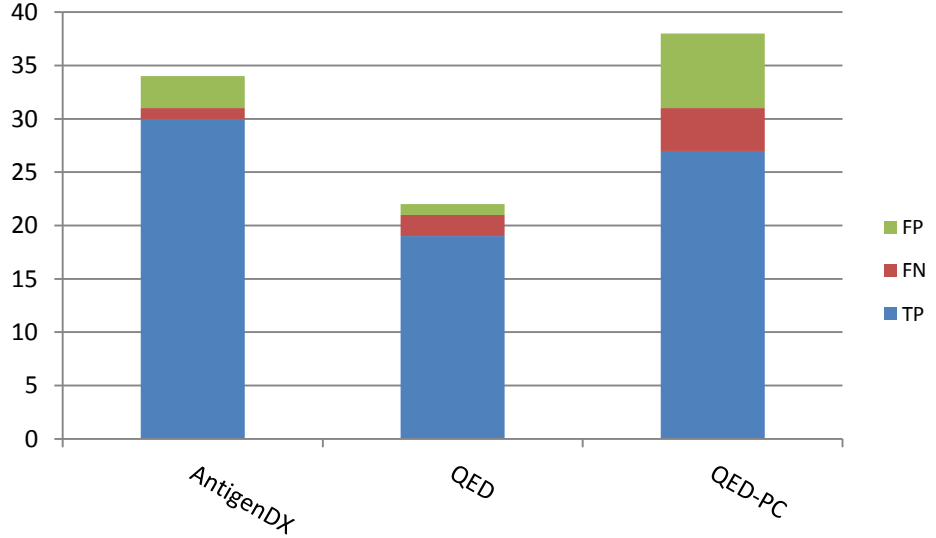
DP-I cost breakdown by scenario fault type



DP-I classification error by scenario fault type



DP-I cost breakdown by scenario action type



DP-I classification error by scenario detection type

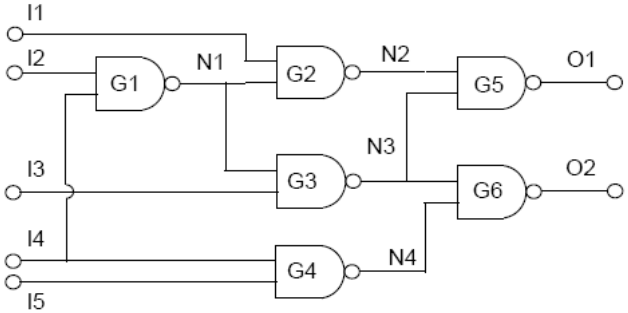
Synthetic Track

Johan de Kleer
Alex Feldman

Synthetic Track Systems

	original				
sys	|IN|	|OUT|	|COMPS|	V	C
74182	9	5	19	47	75
74L85	11	3	33	77	118
74283	9	5	36	81	122
74181	14	8	65	144	228
c432	36	7	160	356	1028
c499	41	32	202	445	1428
c880	60	26	383	826	2224
c1355	41	32	546	1133	3220
c1908	33	25	880	1793	4756
c2670	233	140	1193	2695	6538
c3540	50	22	1669	3388	9216
c5315	178	123	2307	4792	13386
c2688	32	32	2416	4684	14432
c7552	207	108	3512	7232	19312

A simple example ISCAS-85 circuit, c17.



Synthetic Track

What is a difficult observation (cf. challenges)

- high-cardinality faults
- large ambiguity groups
- large number of components

Inject faults of cardinality 1, 2, …, MFMC*

Simplifying assumptions (DXC'11)

- combinational systems - no time (no memory, flip-flops, state, etc)

Large number of observation vectors

Sequential Diagnosis

Synthetic Track Scenario Generation

Every injected fault produces a symptom ('strong' faults)

Every injected fault causally affects the output vector

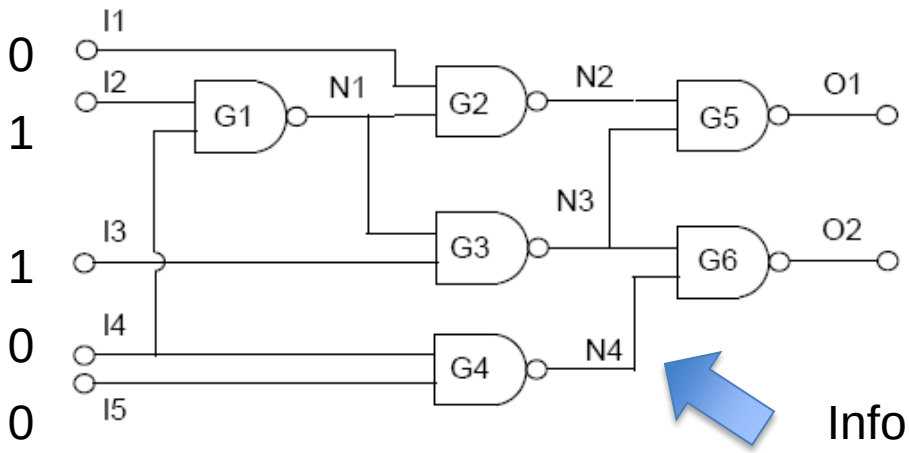
Generate scenarios of ever increasing fault cardinality

- Generate random input vector, compute correct output vector
- Randomly choose output vector bits to flip
- Compute one minc diagnosis
- Continue flipping, following gradient, until input/output pair has desired cardinality
- Generate 10,000 candidates of that ambiguity group
- Randomly chose one as injected fault

Task: Find The Faults by Gathering Information

Inputs fixed given

(Possibly) faulty components



Informative probe

For low cardinality faults, this is almost trivial, but becomes far more difficult at higher cardinality

DA: (1) Requests a probe, (2) reports its diagnoses with their weights.

Synthetic Track Evaluation Metric

$$p + \sum_{\omega \in \Omega} W(\omega)(E[t] + E[\bar{t}])$$

$$E[\bar{t}] = \frac{\bar{n}(\bar{N} + 1)}{\bar{n} + 1}$$

Synthetic Track Results

System	Probee (CPU<0.01)	Optimist (CPU<0.01)	NGDE-NP		NGDE-E				NGDE-F			
	Probes, Score (E=0)	E, Score (Probes=0)	E, Score (Probes=0)	CPU	Probes	E	Score	CPU	Probes	E	Score	CPU
74182	14	14	5.0	0.03	2.5	0	2.5	0.28	2.5	0	2.5	0.02
74L85	30	21	13.8	0.07	3.5	0	3.5	0.22	3.5	0	3.5	0.02
74283	31	25	21.8	0.10	7.4	0	7.4	0.50	7.8	0	7.8	0.07
74181	57	47	35	0.04	8.1	0	8.1	0.55	8.1	0	8.1	0.07
C432	153	118	96	0.08	12.4	0	12.4	0.79	15.8	0	15.8	0.21
C499	170	169	146	1.06	13.9	17.4	31.3	3.1	16.9	0	16.9	0.3
C880	357	306	260	2.2	16.2	38.2	54.4	4.5	17.5	0	17.5	0.3
C1355	514	439	365	4.0	11.8	88.5	100.3	5.6	13.4	0	13.4	0.36
C1908	855	639	571	0.69	11.6	13.1	24.7	6.6	10.2	0	10.2	0.89
C2670	1129	1043	705	3.5	12.5	258	270.5	6.3	14.7	12.9	27.6	1.3
C3540	1647	1094	602	0.47	7.6	0	7.6	2.24	3.2	0	3.2	0.47
C5315	2184	1969	1464	2.9	14	319.8	333.8	6.9	12.5	0	12.5	0.71
C6288	2384	1208	1149	7.4	5.25	443.9	449.15	10.6	6.4	331	337.4	6.4
C7552	3405	2244	1725	1.1	4.2	0	4.2	2.0	2.6	0	2.6	0.98

Caveats: (1) 20 second CPU max, (2) Did not use DXC Framework

Most are trivial, except when very high cardinality (as many as 20) or when large ambiguity group.

Software Track

Alberto Gonzalez-Sanchez
Rui Abreu

Software Track Systems

Program	LOC	Tests	Program Type
print_tokens	539	4130	Lexical Analyzer
print_tokens2	489	4115	Lexical Analyzer
replace	507	5542	Pattern Recognition
schedule	397	2650	Priority Scheduler
schedule2	299	299	Priority Scheduler
tcas	174	1608	Altitude Separation
tot_info	398	1052	Information Measure
space	9126	500	ADL Parser
gzip	6708	211	Compressor
sed	9014	184	Stream Editor
grep	13287	809	String Matching
flex	14194	107	Lexer Generator

Siemens benchmark problems

Software Infrastructure Repository (SIR) benchmark problems

C code with test inputs of full branch coverage

DP-IV Faults

Type	Nominal	Faulty
Relational	a < b	a <= b
	a > b	a >= b
	a == b	a != b
Arithmetic	a + b	a - b
	a * b	a / b
Increment	a++	++a
	a--	--a
Indexing	a[i]	a[i+1]
	a[i]	a[i-1]
Assignment	a = b	a;

Faults seeded by randomly altering source code ('mutation')

Faults affect 'functional' statements, not 'static' code

For each program, 25 scenarios with single fault, 25 scenarios with multiple faults

DAs given 30 seconds to select tests, send probe commands, and return a diagnosis

Software Track Evaluation Metrics

M_{probe}	Number of probes made by DA
M_{cd}	Cost of inspecting suspect candidates top to bottom in ranking
M_{mem}	Memory load

DP-IV Results

DA	M_{probe}	M_{cd}	M_{mem}
Ochiai	109200	1138	3114
Raptor-H	26413	862	8804
Raptor-EPS	11056	889	10320

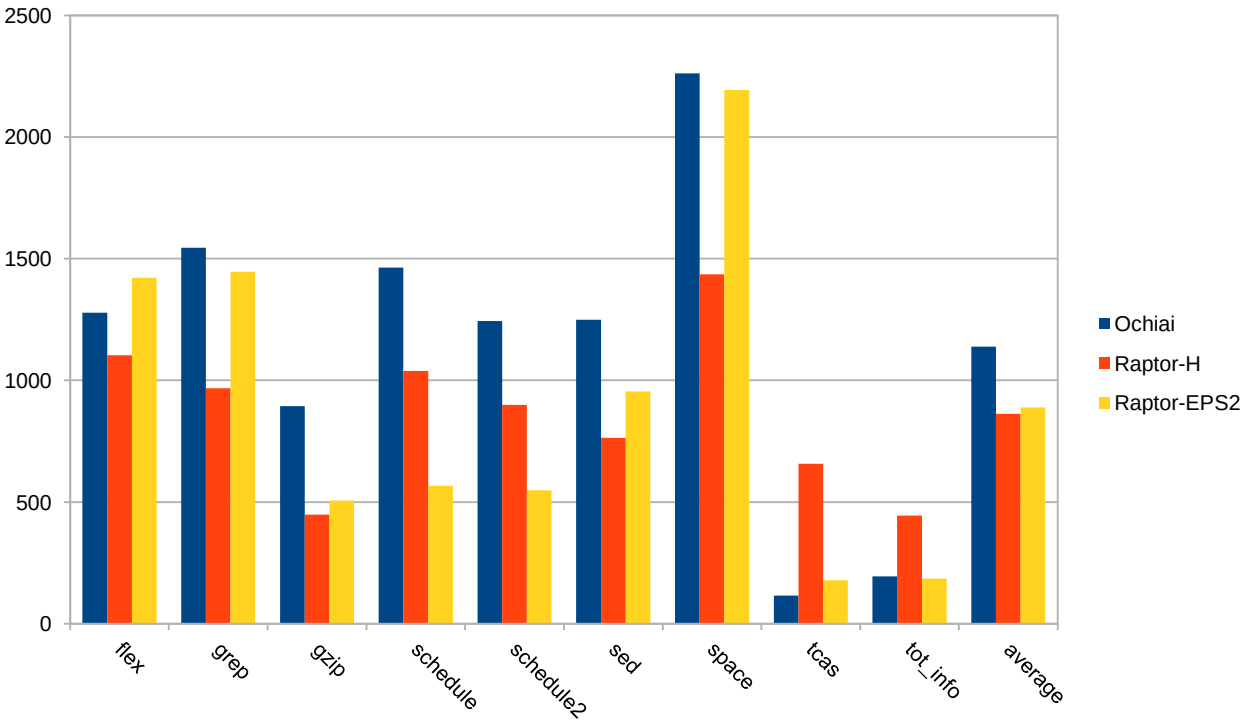
Ochiai requests every possible test, Raptor* includes a test selection heuristic

Raptor-H performs better in scenarios where precomputed intermittency information has low error

Ochiai lighter weight than Raptor* Bayesian diagnosis

DP-IV Results

M_{cd} per program



Raptor-H better than Raptor-EPS2 on large programs

Raptor-EPS2 requires large number of observations, limited by time constraint

Raptor-EPS2 better than Raptor-H on small programs

Future of DXC

Additional DXC organizers? DXC'12 – volunteers?

Common repositories

Benchmarking database

Other Systems

- Proposals?

Acknowledgments

- Shekhar Gupta (PARC)
- Callie Kwiatkowski (University of Wisconsin - Milwaukee)
- David Nishikawa (NASA)
- All DXC'11 competitors
- Many others