# The GCL ANSI Common Lisp Test Suite

Paul Dietz

Motorola Global Software Group

# utline of Talk

- Goal

- Conformance e

# Goals

Primary goal:

Produce a tool for assisting implementors in achieving and maintaining compliance with the ANSI CL standard.

Secondary goals:

- Familiarize myself with the CL standard
- Explore testing methods
- Test the standard itself

N[redacted][redacted]-

# Source

- Harlequin/LispWorks Common Lisp HyperSpec – derived from Ht

# Implementations Tested

- Allegro CL (6.2 and 7.0; x86, Sparc, Power
- Armed Bear Co

Water ’

| Section | Size | Te t | Section | Size | Te t |
|---|---|---|---|---|---|
| Array | 212 | | | | |

# Example of Bug Found: S

# Testing Strategies

- Simple tests .n

Common idiom: confirm some property holds for

# Randomized Test

Myers (in *The Art of Software Testing*):

“Probably the poorest ... methodology of all.

Others have had good results:

- Miller's ‘fuzz testing’
- McKeeman's C com

# Objections to Randomized Testing

- Inefficient
  - Optimize test creation vs. test execution

- Irreproducible
  - Common bugs recur any

## Objections to Randomized Testing (cont.

- Ignores knowledge of program being tested.
  - Knowledge may not be available (black box testing
  - May be wrong or misleading
  - Semi-random tests can incorporate knowledge

# Subtype

- Generate random type $_1$, $_2$.

- If $_1 \subseteq$ $_2$ and S TYPEP ucceed , c

# Compiler Testing

Behavior-prh

# Test of Type Propagation/Inference

- Type inference very useful for efficient lisp compilation.
  - Un[illegible]

# Te ting of T

```
def-ty e- r  -test |+.1| e f  s en
```

```
fun all    m ile nil lambda  )  de lare    timize debug))  symb l
```

## Random Compiler Stress Tester

- Generate random integer-valued form

- Found bugs within seconds in all implementations.
- Most failures were assertion failures, type errors, or incorrect values.
- Bugs that crashed the lisp were infrequent.
- Many dead code, type inference bugs.

# Experience ith CLIS

# Automated Pruning

- Form

Comment

```
 TYPE-█F 17) ==> FIXN █M

I ██N
```

No!

# A Problem V

Que tion ?