FORTRAN

Bugs I have known and loved!

Ron Bell
ron.bell@awe.co.uk
1966 – 1970

- Oxford University research student at AERE Harwell
- 1966: decks of 80-column punched cards couriered twice a day to the IBM Stretch at AWRE Aldermaston
  - Highly idiosyncratic compiler
    - COMMON
    - SUBROURINEs
- 1966-70: IBM 360/65 at AERE
  - Still 80-column punched cards
1970-2003
- Worked for IBM
- 1970-1990 Systems Engineer
  - Scientific customers – AERE Harwell etc.
  - Mainframes (System 360/370)
  - FORTRAN G, H compilers, etc.
  - The horrors of IBM JCL:
    - //GO.FT01F001 DD
    - DSN=OUTPUT,UNIT=3330,DISP=(NEW,KEEP),SPACE=(CYL,(5,1)),........

1990-2003
- HPC and FORTRAN Specialist
- IBM RS/6000 and SP.
- XL FORTRAN, MPI etc.
2003 - now

AWE, Aldermaston
HPC Optimisation Consultant
“Blue Oak”
  IBM SP (16-way Nighthawk nodes)
  1856 PEs
  2.9 Tflops peak
  IBM XL FORTRAN

Now moving to REDWOOD
  Cray XT3 (dual core 2.6 GHz Opteron nodes)
  7888 PEs
  41 Tflops peak
  PGI FORTRAN
A selection of 3 Interesting and Amusing Bugs

- The Amazing Value Bug
- The Electricity Generation Bug
- The DGEMM Performance Bug

None of these involves memory over-write due to overflowing array bounds
The Amazing Value Bug

- On AERE 360/65, IBM FORTRAN G
- After a lot of investigation:

\[
\begin{align*}
\text{A} & = 1.0 \\
\text{PRINT 1,A} \\
1 & \quad \text{FORMAT(1H,F10.3)} \\
\end{align*}
\]

- Resultant printout
  - 34.176
- This was not a compiler bug!
The Electricity Generation Bug

- Converting code from IBM mainframe to RS/6000 and Sun
- Executable on mainframe gave correct reference answers
  - Relied on for years in electricity industry
- Initial run on RS/6000 gave wrong answers
- Recompilation of source code on mainframe gave same wrong answers as on RS/6000
- Mainframe executable had been compiled at OPT(3)
- Recompile on mainframe at OPT(3) – answers now correct
- Try RS/6000 at OPT(3) – same wrong answers as OPT(0)

- Wrong answers: RS6K at OPT(0) and (3). Mainframe at OPT(0)
  - All wrong answers identical
- Correct answers: Mainframe at OPT(3)
I presented to a customer claiming DGEMM in IBM’s ESSL would give sustained performance on RS/6000 over 90% of peak on large matrices.

After a few days, a bright spark from the customer rang me to say he was getting less than 50%.
The Amazing Value Bug

- On AERE 360/65, IBM FORTRAN G
- After a lot of investigation:

  ```fortran
  A = 1.0
  PRINT 1, A
  1 FORMAT(1H ,F10.3)
  .
  
  Resultant printout
  - 34.176
  
  This was not a compiler bug!
The Amazing Value Bug - SOLUTION

CALL SUBX(1.0)

A = 1.0
PRINT 1,A

SUBROUTINE SUBX(Z)

Z = 34.176

RETURN
END

Illegal FORTRAN – user error.
Not compiler bug! - though compilers overcome this nowadays!
The Electricity Generation Bug

- Converting code from IBM mainframe to RS/6000 and Sun
- Executable on mainframe gave correct reference answers
  - Relied on for years in electricity industry
- Initial run on RS/6000 gave wrong answers
- Recompilation of source code on mainframe gave same wrong answers as on RS/6000
- Mainframe executable had been compiled at OPT(3)
- Recompile on mainframe at OPT(3) – answers now correct
- Try RS/6000 at OPT(3) – same wrong answers as OPT(0)

Wrong answers: RS6K at OPT(0) and (3). Mainframe at OPT(0)
  - All wrong answers identical
Correct answers: Mainframe at OPT(3)
The Electricity Generation Bug - SOLUTION

INTEGER*2 IA, IB, IC

IA = IB * IC
CALL SUBY(IA)

- Integer multiplication overflowed
- RS/6000 correctly passed overflowed value to the subroutine at all optimisation levels
- Mainframe at OPT(0) stored overflowed value in IA before loading into (INTEGER*4) register to pass to subroutine
- At OPT(3) result of IB*IC in register with non-overflowed value passed directly to subroutine – no need to store in IA

CONCLUSION: Bug in mainframe compiler at OPT(3) gave results correct as programmer intended – but wrong according to what he actually coded.
The DGEMM Performance Bug

- I presented to a customer claiming DGEMM in IBM’s ESSL would give sustained performance on RS/6000 over 90% of peak on large matrices.
- After a few days, a bright spark from the customer rang me to say he was getting less than 50%.
- ????????
The perils of FORTRAN 90!!

CALL DGEMM (‘N’,’N’,N,N,N,1D0,A(:,,:),N,B(:,,:),N,0D0,C(:,,:),N)

Either of following was OK:

CALL DGEMM (‘N’,’N’,N,N,N,1D0,A(1,1),N,B(1,1),N,0D0,C(1,1),N)
CALL DGEMM (‘N’,’N’,N,N,N,1D0,A,N,B,N,0D0,C,N)

- A(:, :) is a Fortran 90 array section
- DGEMM required a contiguous 2D array
- Early F90 compiler reckoned it needed to make a copy of any array section to ensure it was contiguous
- The copy took nearly as long as DGEMM