Sustainable and Reliable Computing with Tools: Analyzing Precision Appetites of CFD Applications with VerifiCarlo

Roman Iakymchuk[1]
joint work with Pablo Oliveira (Paris-Saclay)

Umeå University, Sweden
riakymch@cs.umu.se

MS4F - Cross-Cutting Aspects of Exploiting Exascale Platforms for High-Fidelity CFD in Turbulence Research
PASC, Davos, Switzerland
June 27th, 2023

[1]This research is partially funded by EuroHPC JU CoE CEEC (No. 101093393)

Outline

1. Floating-point arithmetic
2. Robustness of algorithms
3. An approach toward sustainable computing

UMEÅ UNIVERSITY

Floating-point arithmetic (1/2)

Computer arithmetic approximates real numbers with their finite representations

Floating-point arithmetic (1/2)

Computer arithmetic approximates real numbers with their finite representations

Issues

- Floating-point arithmetic suffers from rounding errors
- Floating-point operations $(+, \times)$ are commutative but non-associative

$$(-1+1)+2^{-53} \neq -1+(1+2^{-53}) \quad \text{in double precision}$$

UMEÅ UNIVERSITY

Floating-point arithmetic (1/2)

Computer arithmetic approximates real numbers with their finite representations

Issues

- Floating-point arithmetic suffers from rounding errors
- Floating-point operations $(+, \times)$ are commutative but non-associative

$$2^{-53} \neq 0 \quad \text{in double precision}$$

UMEÅ UNIVERSITY

Floating-point arithmetic (1/2)

Computer arithmetic approximates real numbers with their finite representations

Issues

- Floating-point arithmetic suffers from rounding errors
- Floating-point operations $(+, \times)$ are commutative but non-associative

$$(-1+1)+2^{-53} \neq -1+(1+2^{-53}) \quad \text{in double precision}$$

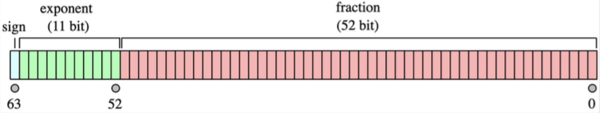
- Consequence: results of floating-point computations depend on the order of computation
- Results computed by performance-optimized parallel floating-point libraries may be often inconsistent

UMEÅ UNIVERSITY

Floating-point arithmetic (2/2)

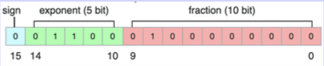
Almost all computer hardware and software support the **IEEE Standard for Floating-Point Arithmetic** IEEE 754-2019

- double - binary64

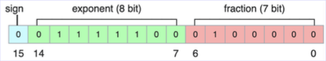


- single - binary32

- half - binary16

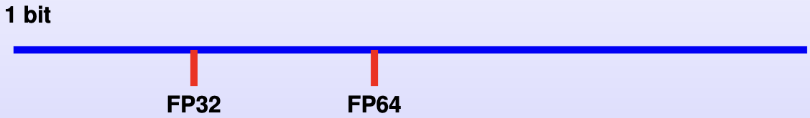


- bfloat



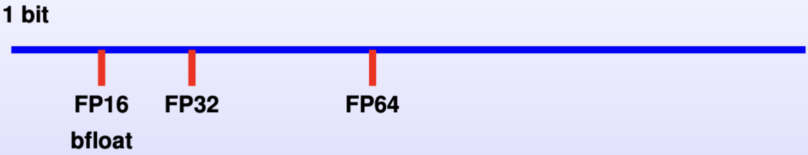
UMEÅ UNIVERSITY

Mixed-precision arithmetic



- double-single plus iterative refinement

Mixed-precision arithmetic

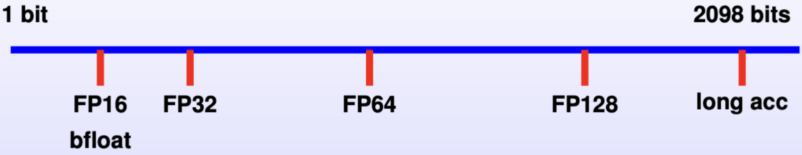


- double-single plus iterative refinement
- double-single-half/ bfloat
- Over 100 works on mixed precision [a]

---

[a] Nicholas Higham and Théo Mary. 'Mixed Precision Algorithms in Numerical Linear Algebra'

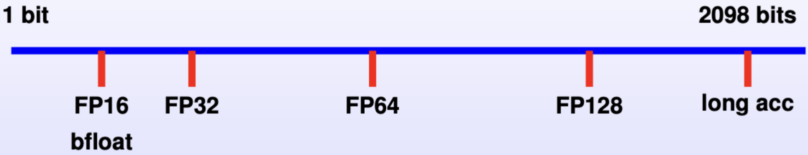
Mixed-precision arithmetic



- double-single plus iterative refinement
- double-single-half/ bfloat
- Over 100 works on mixed precision
- Extending precision for exact computations: double plus FPEs or double plus long accumulator

UMEÅ UNIVERSITY

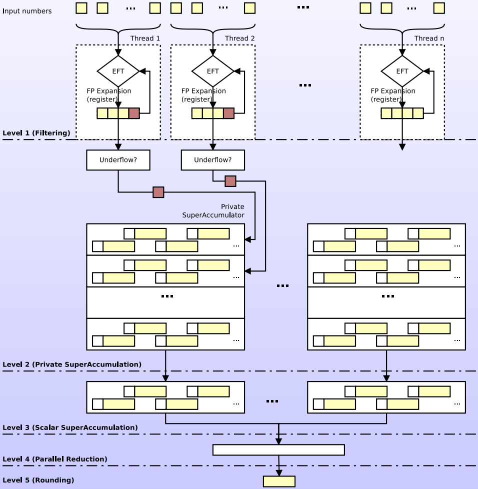
Mixed-precision arithmetic



- double-single plus iterative refinement
- double-single-half/ bfloat
- Over 100 works on mixed precision
- Extending precision for exact computations: double plus FPEs or double plus long accumulator
- **Mixed-precision algorithm** is an algorithm that carefully, effectively and safely combines multiple precisions

ExBLAS: Parallel Reduction

Highlights of the Algorithm



[a]

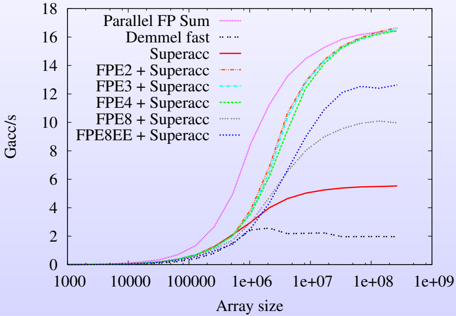
- Based on FPE with EFT and Kulisch accumulator
- Suitable for CPUs, GPUs, Xeon Phi
- Guarantees "inf" precision

→ **bit-wise reproducibility**

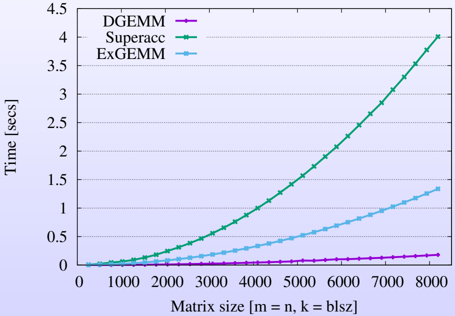
[a] S. Collange, R. Iakymchuk et al. Numerical Reproducibility for the Parallel Reduction on Multi- and Many-Core Architectures. ParCo, 49, 2015, 83-97

Performance results

sum: $\alpha := \sum_i^N x_i$



gemm: $C := AB$



- Correctly rounded and reproducible
- Tiny performance overhead

- Correctly rounded and reproducible
- 10x-25x overhead

- Our approach performs well on memory-bound computations

$$\boxed{Ax = b}$$

while $(\tau > \tau_{\max})$

Step	Operation	Kernel	Communication
$S1:$	$w := Ad$	SPMV	Alltoallw*
$S2:$	$\rho := \beta / < d, w >$	DOT product	Allreduce
$S3:$	$x := x + \rho d$	AXPY	–
$S4:$	$r := r - \rho w$	AXPY	–
$S5:$	$z := M^{-1} r$	Apply preconditioner	–
$S6:$	$\beta := < z, r >$	DOT product	Allreduce
$S7:$	$d := (\beta / \beta_{old}) d + z$	AXPY-like	–
$S8:$	$\tau := < r, r >$	DOT product	Allreduce

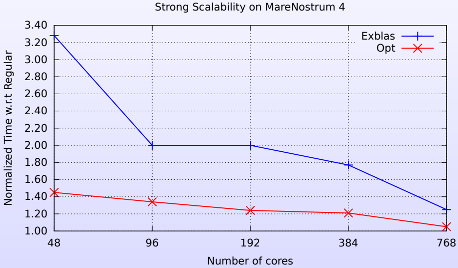
end while

Robustness of Algorithms

- **Robustness: accuracy and reproducibility**
- FP ops are non-associative : $(-1+1)+2^{-53} \neq -1+(1+2^{-53})$
- Non-reproducibility in PCG: `dot`, `axpy`, and `spmv`

→ Solution : ExBLAS (ParCo15, NRE15, JCAM, IJHPCA)

3D Poisson equation with 27 stencil points and $tol = 10^{-8}$



Iteration	Residual			
	MPFR	*Original* 1 proc	*Original* 48 procs	*Exblas & Opt*
0	0x1.19f179eb7f032p+49	0x1.19f179eb7f03**3**p+49	0x1.19f179eb7f03**3**p+49	0x1.19f179eb7f032p+49
2	0x1.f86089ece9f75p+38	0x1.f86089**f08810d**p+38	0x1.f86089e**d07a76**p+38	0x1.f86089ece9f75p+38
9	0x1.fc59a29d329ffp+28	0x1.fc59a29d**1b6a**p+28	0x1.fc59a29d**2e989**p+28	0x1.fc59a29d329ffp+28
10	0x1.74f5ccc211471p+22	0x1.74f5cc**b8203ad**p+22	0x1.74f5ccc**1fafef**p+22	0x1.74f5ccc211471p+22
...	...	...	...	...
40	0x1.7031058eb2e3ep-19	0x1.703105**aea0e8a**p-19	0x1.7031058e**8ff5a**p-19	0x1.7031058eb2e3ep-19
42	0x1.4828f76bd68afp-23	0x1.4828f**6fabbf2a**p-23	0x1.4828f76b**b9038**p-23	0x1.4828f76bd68afp-23
45	0x1.8646260a70678p-26	0x1.86462601**300d2**p-26	0x1.8646260a7**1301**p-26	0x1.8646260a70678p-26
47	0x1.13fa97e2419c7p-33	0x1.13fa**98038c44e**p-33	0x1.13fa97e**54e903**p-33	0x1.13fa97e2419c7p-33

Table 3: Accuracy and reproducibility comparison on the intermediate and final residual against MPFR for a matrix with condition number of 10^{12}. The matrix is generated following the procedure from Section 5.1 with n=4,019,679 (159^3).

European Processor Initiative

VRP and STX

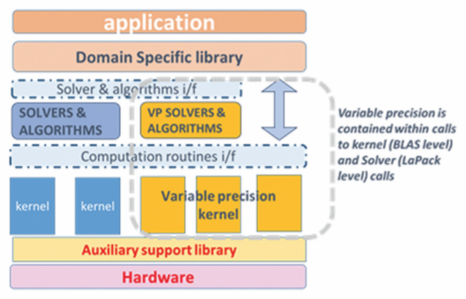
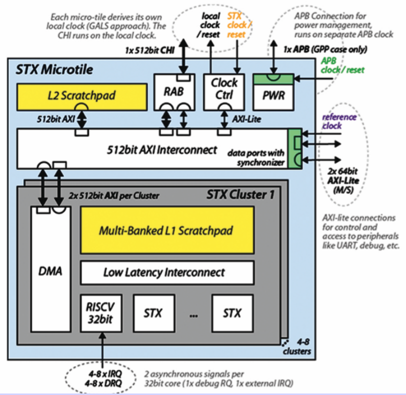


Fig: Layered programming model

- for large ill-conditioned systems
- "when the standard precision unit cannot reach the expected accuracy, the variable precision unit takes the relay"
- zero-copy from GPP to VRP



- Stencil/ tensor accelerator
- Energy efficiency
- Posit-based ML & DNN Acceleration

Source: EPI

Sustainable HPC → Energy-efficient HPC

- **Energy-efficient architectures** such as graphic processors (GPUs) and FPGAs – Green HPC computing
- PDC@KTH extracts the produced heat to **warm up the main campus**
- CSCS at Switzerland proposes **‘free cooling’** with the water from the lake of Lugano

UMEÅ UNIVERSITY

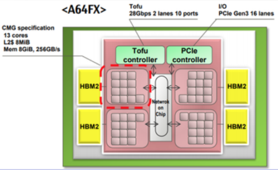
Precision & Sustainability

Exascale computing and linear algebra

- Exascale computing is constrained by **power consumption**

→ Power-efficient hardware

  - RIKEN's Fugaku w A64FX (FP64:FP32:FP16 = 1:2:4)
  - EPI (ARM, FPGA, RISC-V)



Source: Fujitsu

- Linear algebra is known to be dominant by **double precision**

→ Sustainable algorithms

math **Mixed and adaptive precision computing**

code **Communication hiding or avoiding**

tools **Numerical abnormalities and precision cropping**

UMEÅ UNIVERSITY

Robust and sustainable algorithms

Idea

lagom - not too much, not too little, *just the right amount*

Robust and sustainable algorithms

Idea

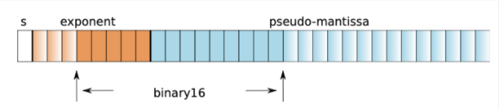
lagom - not too much, not too little, *just the right amount*

Analysis w tools → Strategy → Revision of algorithms

1. **Arithmetic tool** applied to **code** → manual/ automatic
2. if the reduction is possible, apply algorithmic solutions
3. conduct probabilistic (aka optimistic) error analysis
   - error bound with constant $\sqrt{n}\mu$ with high probability
4. implement on hardware with stochastic rounding support – randomly maps x to one of two bounds

Analysis with tools: VerifiCarlo

- Verificarlo – an automatic tool for debugging and assessing FP precision based on Monte Carlo Arithmetic
- Backends: debugging (MCA) and mixed-precision (Vprec)
- Eg setting $r = 5$ and $p = 10$, Vprec simulates a `binary16` embedded inside a `binary32`

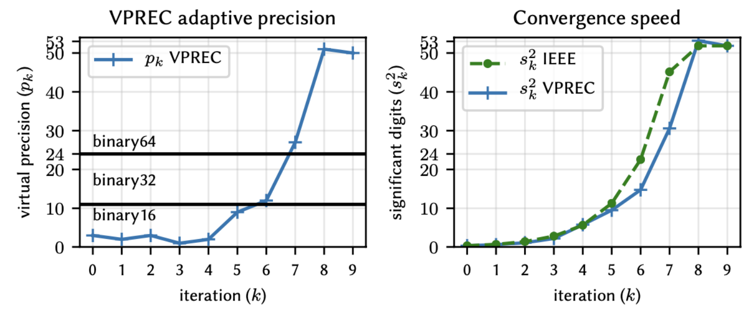


k	x_{k+1}	s_k^{10}	s_k^2
0	0.0690266447076745	0.11	0.37
1	0.1230846130203958	0.21	0.70
2	0.1985746566605835	0.43	1.43
3	0.2732703639721015	0.84	2.79
4	0.3119369815109966	1.79	5.95
5	0.3181822938100336	3.40	11.3
6	0.3183098350392471	6.79	22.6
7	0.3183098861837825	13.6	45.2
8	0.3183098861837907	15.6	51.8
9	0.3183098861837907	15.6	51.8

```
double newton(double x0) {
  double x_k, x_k1=x0, b=PI;
  do {
    x_k  = x_k1;
    x_k1 = x_k*(2-b*x_k);
  }while (fabs((x_k1-x_k)/x_k)
  >= 1e-15);
  return x_k1;
}
```

The Newton-Raphson method for inverse of π[a]

[a]Pablo Oliveira et al. *Automatic exploration of reduced floating-point representations in iterative methods.* Euro-Par 2019

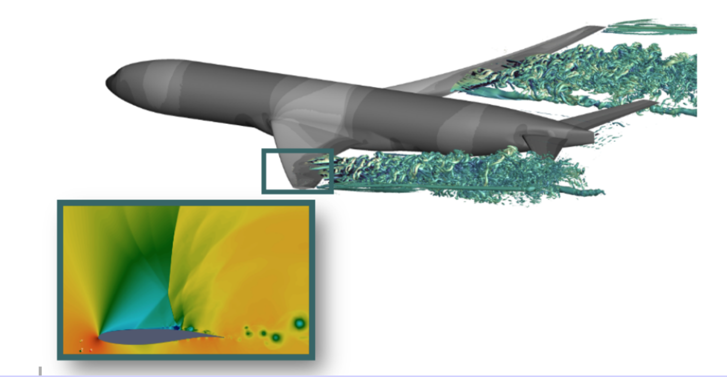


The Newton-Raphson method for inverse of π[a]

[a] Pablo Oliveira et al. *Automatic exploration of reduced floating-point representations in iterative methods.* Euro-Par 2019

EuroHPC JU CoE CEEC: 2023-26

Center of Excellence for Exascale CFD



- **Sustainable algorithmic solutions with mixed-precision and tools**
- **Consortium codes**: Neko, NEK5000/ NekRS, FLEXI, waLBerla

Test cases

- **NEK5000** is a high order, incompressible Navier-Stokes solver based on the spectral element method
- **Nekbone** solves a Poisson equation using a Conjugate Gradient method with a simple or spectral element multigrid preconditioner
- **AMG** benchmark – parallel algebraic multigrid solver for linear systems arising from problems on unstructured grids
  - Two solvers: CG and GMRES

UMEÅ UNIVERSITY

Nekbone w `Vprec`

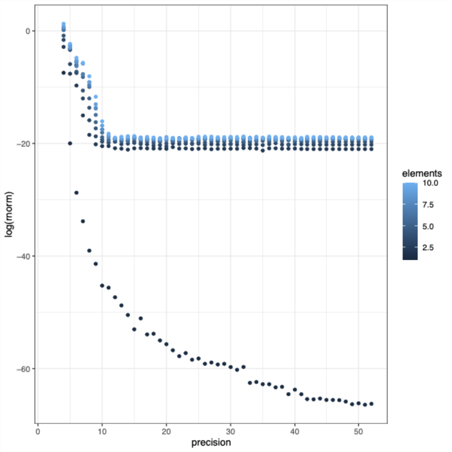
Basic example w/o preconditioner

$$\boxed{Ax = b}$$

while $(\tau > \tau_{\max})$

Step	Operation
$S1:$	$w := Ad$
$S2:$	$\rho := \beta / < d, w >$
$S3:$	$x := x + \rho d$
$S4:$	$r := r - \rho w$
$S5:$	$z := M^{-1} r$
$S6:$	$\beta := < z, r >$
$S7:$	$d := (\beta / \beta_{old}) d + z$
$S8:$	$\tau := < r, r >$

end while



Nekbone w `Vprec`

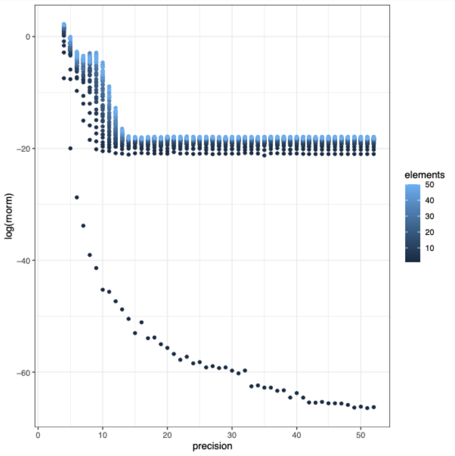
Multigrid Preconditioner Example

$$\boxed{Ax = b}$$

while $(\tau > \tau_{\max})$

Step	Operation
$S1:$	$w := Ad$
$S2:$	$\rho := \beta / < d, w >$
$S3:$	$x := x + \rho d$
$S4:$	$r := r - \rho w$
$S5:$	$z := M^{-1} r$
$S6:$	$\beta := < z, r >$
$S7:$	$d := (\beta / \beta_{old}) d + z$
$S8:$	$\tau := < r, r >$

end while



Nekbone w MCA

Multigrid Preconditioner Example: PCG

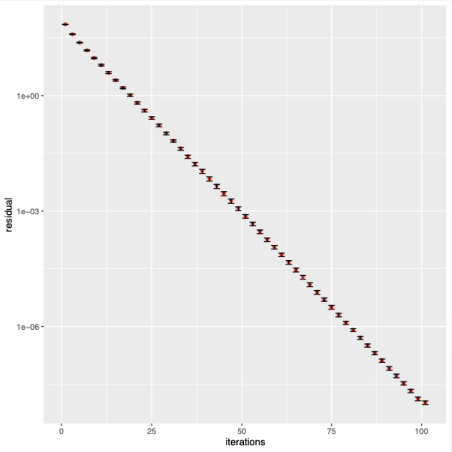
- **20 MCA samples** for the previously found Vprec configuration
- simulate `binary32`

$$\boxed{Ax = b}$$

while $(\tau > \tau_{\max})$

Step	Operation
$S1:$	$w := Ad$
$S2:$	$\rho := \beta / < d, w >$
$S3:$	$x := x + \rho d$
$S4:$	$r := r - \rho w$
$S5:$	$z := M^{-1} r$
$S6:$	$\beta := < z, r >$
$S7:$	$d := (\beta / \beta_{old}) d + z$
$S8:$	$\tau := < r, r >$

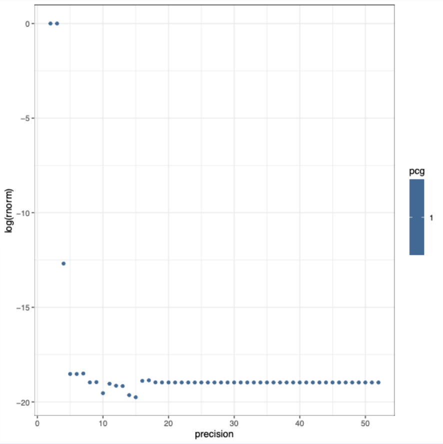
end while



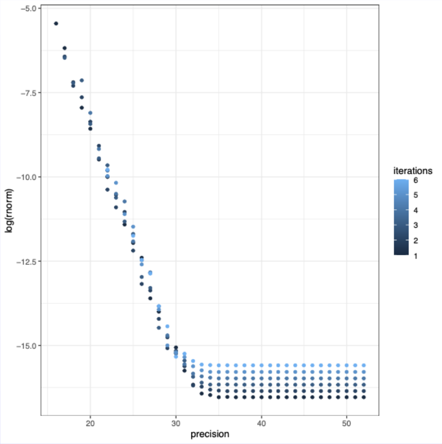
UMEÅ UNIVERSITY

AMG w Vprec

AMG-PCG
Laplace type problem



AMG-GMRES
Non-linear time-dependent problem



UMEÅ UNIVERSITY

Summary

- mixed-precisions is a way toward sustainable computing
- employ computer arithmetic tools for
  - numerical abnormalities detection
  - precision requirements inspection
  - numerical CI etc

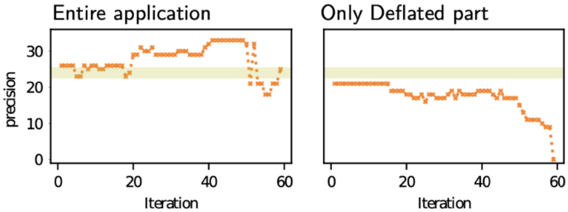


Figure: Minimal precision that preserves convergence.

Yales2 CFD application with the DPCG solver: 16 % energy gain

Automatic exploration of reduced floating-point representations in iterative methods.

Chatelain, Petit, de Oliveira Castro, Lartigue, Defour. Euro-Par 2019

Summary

- mixed-precisions is a way toward sustainable computing
- employ computer arithmetic tools for
  - numerical abnormalities detection
  - precision requirements inspection
  - numerical CI etc

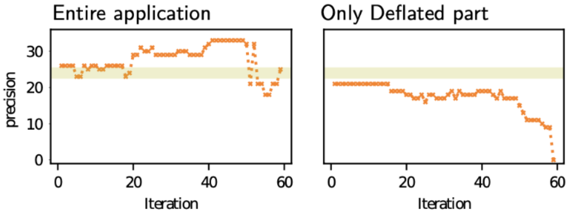


Figure: Minimal precision that preserves convergence.

Yales2 CFD application with the DPCG solver: 16 % energy gain

Automatic exploration of reduced floating-point representations in iterative methods.

Chatelain, Petit, de Oliveira Castro, Lartigue, Defour. Euro-Par 2019

Thank you for your attention !

UMEÅ UNIVERSITY