Benchmarks:

- Basically two different goals
- Users: how fast will my app run?
- Researchers: how good is my...?

```
large      real app-based
          \   `/    `
           \     `/    `
small    micro benchmarking
```

\[
\text{App-based} \rightarrow \{ \text{SPEC CPU, SPEC WEP, SPEC JBB (for Java), TPC (database benchmarks)} \}
\]

Rarely used in academia

Compile benchmarks

Compile benchmarks are CPU bound

\* It is important that benchmark captures the item under consideration with minimum possible noise.

Kernel, synthetic, trace replay: - postmark, bonnie, bonnie++, iometer

Synthetic benchmarks are a set of steps that represent some real-life system.

We can tune these tools to simulate any desired workload or environment so as to make it as close as possible to the real system.
Trans-runtime software benchmarking is generally used in cases where you want to simulate an older run of a system and it is good enough in some sense that it captures all the important attributes.

\[\text{real system} \xrightarrow{\text{trace}} \text{replay} \xrightarrow{} \text{system under test}\]

\textit{e.g.}

\[
\begin{align*}
\text{user} & \quad \rightarrow \quad t_2 \leftarrow \\
\text{interrupt} & \quad \rightarrow \quad T_i \\
\end{align*}
\]

Interrupt occurring at the end or at the middle of a running process.
Micro-benchmarks:

- ping-pong context switch

fork, exec benchmarks

Other interesting thing:

There is a survey paper from researchers in Stony Brook with regard to systems papers published in last 5-10 years.

There is a interesting paper from last year that talks about things like new benchmark in optimization level on effect space for SPEC suite results.