Towards Predicting the Impact of Software Changes on Building Activities

M. Tufano, H. Sajnani, K. Herzig
Continuous Integration
Continuous Integration

Developers build daily.
Continuous Integration

Developers build daily. Many times a day.
(some) Microsoft Builds
(some) Microsoft Builds

2,000 builds per hour.
Builds in the Cloud

Distributed and parallel builds on remote cloud infrastructures

Microsoft
CloudBuild

Google
Bazel

Facebook
Buck
CloudBuild

Distributed builds on many machines in the cloud

Parallelized build tasks

Content-based cache to accelerate builds

Builds, test, code analysis, drops, package, and storage.
Faster Builds

Build time is the bottleneck for shipping faster.

Not only improvements on the infrastructure side. Attention to developers’ changes.
Faster Builds

Build time is the bottleneck for shipping faster.

Not only improvements on the infrastructure side.
Attention to developers’ changes.

Focus of this paper!
Faster Builds

Developer Changes
- Dependencies
- Architectural

Early Feedback
- Awareness of impact
- Early restructuring
- Avoid build time regression
Predicting the **impact** is challenging with modern **cached** build systems.
Build Dependency Graph

Build Target

Dependency

Dependent
Cached Build System

12%  5%  60%

Probability Target is built (not from cache)

Frequent LCP
Cached Build System

12%  5%  60%

Added Target

Probability
Target is built
(not from cache)

This change will
have a greater
impact than this...
Impact of Software Changes on Building Activities

Predict Impact
- Build Time increase
- Percentage of future builds affected

Approximation using Telemetry data
- Probabilities
- Target Execution Time
Software Changes introducing dependencies

- New Outward dependency from LCP

- New Inward dependency to LCP

Frequent LCPs from recent build activities
Standards

\[ t_0, t_1, \ldots, t_n \]  

Order of execution

Functions

\[ \text{ExecTime}(t_0, \ldots, t_i) \] Estimation of execution time of a sequence of targets

\[ \text{BuildCoverage}(t_i) \] Estimation of percentage of builds building the target (not from cache)
Outward Dependency

New dependency added \((t_i \rightarrow x)\) from a LCP node

\[ DC_2 \]

\[ DC_1: \text{LCP} \]
Outward Dependency

New dependency added \((t_i \rightarrow x)\) from a LCP node

\(DC_2\)

\(DC_1: LCP\)

After the change:

\[
\text{IF } \text{ExecTime}(s_0, \ldots, x) \leq \text{ExecTime}(t_0, \ldots, t_{i-1}) \\
\text{LCP}: DC_1 \text{ (unchanged)}
\]
Outward Dependency

New dependency added \((t_i \rightarrow x)\) from a LCP node

\[ DC_2 \]

\[ DC_1: LCP \]

After the change:

\[
\text{IF } \text{ExecTime}(s_0, \ldots, x) \leq \text{ExecTime}(t_0, \ldots, t_{i-1}) \\
\quad LCP: DC_1 \text{ (unchanged)}
\]

\[
\text{ELSE IF } \text{ExecTime}(s_0, \ldots, x) > \text{ExecTime}(t_0, \ldots, t_{i-1}) \\
\quad LCP: DC_3: (s_0, \ldots, x, t_i, \ldots, t_n)
\]
Outward Dependency

New dependency added \((t_i \rightarrow x)\) from a LCP node

\[ DC_2 \]

\[ DC_1: LCP \]

After the change:

\[
\begin{align*}
\text{IF } \text{ExecTime}(s_0, \ldots, x) \leq \text{ExecTime}(t_0, \ldots, t_{i-1}) \\
\text{LCP} : DC_1 \text{ (unchanged)}
\end{align*}
\]

\[
\begin{align*}
\text{ELSE IF } \text{ExecTime}(s_0, \ldots, x) > \text{ExecTime}(t_0, \ldots, t_{i-1}) \\
\text{LCP} : DC_3 : (s_0, \ldots, x, t_i, \ldots, t_n)
\end{align*}
\]

Change Impact:

\[
\begin{align*}
\text{Time increase: } \text{ExecTime}(s_0, \ldots, x) - \text{ExecTime}(t_0, \ldots, t_{i-1}) \\
\text{Percentage of affected builds: } \text{BuildCoverage}(x)
\end{align*}
\]
Inward Dependency

New dependency added \((x \rightarrow t_i)\) to a LCP node

\[ DC_2 \]

\[ DC_1: LCP \]
Inward Dependency

New dependency added $(x \rightarrow t_i)$ to a LCP node

$DC_2$

$DC_1$: LCP

After the change:

\[
\text{IF } \text{ExecTime}(s_0, \ldots, s_j) \geq \text{ExecTime}(t_0, \ldots, t_i) \\
\text{LCP: } DC_1 \text{ (unchanged)}
\]
Inward Dependency

New dependency added \((x \rightarrow t_i)\) to a LCP node

\[DC_2\]

\[DC_1: \text{LCP}\]

After the change:

\[\text{IF } \text{ExecTime}(s_0, \ldots, s_j) \geq \text{ExecTime}(t_0, \ldots, t_i)\]

\[\text{LCP: } DC_1 \text{ (unchanged)}\]

\[\text{ELSE IF } \text{ExecTime}(s_0, \ldots, s_j) < \text{ExecTime}(t_0, \ldots, t_i)\]
Inward Dependency

New dependency added \((x \rightarrow t_i)\) to a LCP node

\[ DC_2 \]

\[ Tier_0 \quad s_0 \quad s_j \quad s_m \quad Tier_{j+1} \quad Tier_m \]

\[ DC_1: LCP \]

\[ Tier_0 \quad t_0 \quad t_i \quad t_n \quad Tier_n \]

After the change:

IF \( \text{ExecTime}(s_0, \ldots, s_j) \geq \text{ExecTime}(t_0, \ldots, t_i) \)

\[ LCP: DC_1 \text{ (unchanged)} \]

ELSE IF \( \text{ExecTime}(s_0, \ldots, s_j) < \text{ExecTime}(t_0, \ldots, t_i) \)

\[ \text{IF } \text{ExecTime}(x, \ldots, s_m) \leq \text{ExecTime}(t_{i+1}, \ldots, t_n) \]

\[ LCP: DC_1 \text{ (unchanged)} \]
Inward Dependency

New dependency added \((x \rightarrow t_i)\) to a LCP node

\[
\begin{align*}
DC_2 & \quad s_0 \rightarrow s_j \rightarrow x \rightarrow s_m \\
DC_1: LCP & \quad t_0 \rightarrow t_i \rightarrow \ldots \rightarrow t_n
\end{align*}
\]

Tier_0 \quad Tier_j \quad Tier_{j+1} \quad Tier_m

Tier_0 \quad Tier_i \quad Tier_n

After the change:

\[
\begin{align*}
\text{IF } & \text{ExecTime}\left(s_0, \ldots, s_j\right) \geq \text{ExecTime}\left(t_0, \ldots, t_i\right) \\
& \quad LCP: DC_1 \text{ (unchanged)} \\
\text{ELSE IF } & \text{ExecTime}\left(s_0, \ldots, s_j\right) < \text{ExecTime}\left(t_0, \ldots, t_i\right) \\
& \quad \quad \text{IF } \text{ExecTime}\left(x, \ldots, s_m\right) \leq \text{ExecTime}\left(t_{i+1}, \ldots, t_n\right) \\
& \quad \quad \quad \quad LCP: DC_1 \text{ (unchanged)} \\
& \quad \quad \quad \text{ELSE IF } \text{ExecTime}\left(x, \ldots, s_m\right) > \text{ExecTime}\left(t_{i+1}, \ldots, t_n\right) \\
& \quad \quad \quad \quad LCP: DC_2: \left(t_0, \ldots, t_i, x, \ldots, s_m\right)
\end{align*}
\]
**Inward Dependency**

New dependency added \((x \rightarrow t_i)\) to a LCP node

**DC\(_2\)**

\[ s_0 \rightarrow s_i \rightarrow x \rightarrow s_m \]

**DC\(_1\): LCP**

\[ t_0 \rightarrow t_i \rightarrow t_{n} \]

After the change:

**IF** \(\text{ExecTime}(s_0, \ldots, s_i) \geq \text{ExecTime}(t_0, \ldots, t_i)\)

\(\text{LCP: } DC_1\) (unchanged)

**ELSE IF** \(\text{ExecTime}(s_0, \ldots, s_i) < \text{ExecTime}(t_0, \ldots, t_i)\)

**IF** \(\text{ExecTime}(x, \ldots, s_m) \leq \text{ExecTime}(t_{i+1}, \ldots, t_n)\)

\(\text{LCP: } DC_1\) (unchanged)

**ELSE IF** \(\text{ExecTime}(x, \ldots, s_m) > \text{ExecTime}(t_{i+1}, \ldots, t_n)\)

\(\text{LCP: } DC_2:\) \((t_0, \ldots, t_i, x, \ldots, s_m)\)

**Change Impact:**

Time increase: \(\text{ExecTime}(x, \ldots, s_m) - \text{ExecTime}(t_{i+1}, \ldots, t_n)\)

Percentage of affected builds: \(\text{BuildCoverage}(t_i)\)
Conclusions

Build Time Regression
- Threat for fast software delivery
- Difficult to diagnose and correct

Predict Build Impact
- Provide contextual info during the Pull Request process
- Allow early corrective operations
Future Work

Evaluation
- Run in shadow mode and estimate impact
- Evaluate prediction accuracy

Positive Feedback
- Positive impact on build activities
- Estimate reduction in build time
Questions?

Towards Predicting the Impact of Software Changes on Building Activities

M. Tufano, H. Sajnani, K. Herzig