東邦大学

生命（いのち）の科学で未来をつなぐ

パイプラインのハザード

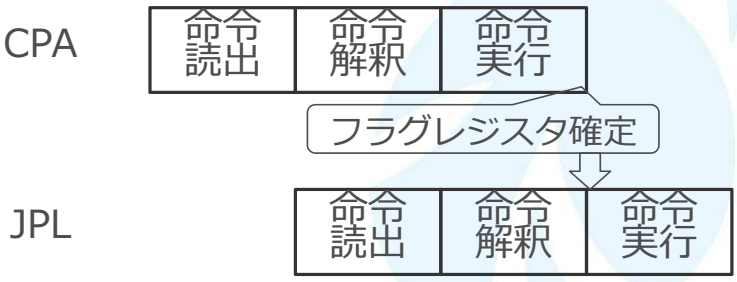
パイプラインのハザード

パイプラインが途切れて
うまく動かないときのこと

パイプラインのハザード

パイプラインが途切れて
うまく動かないときのこと

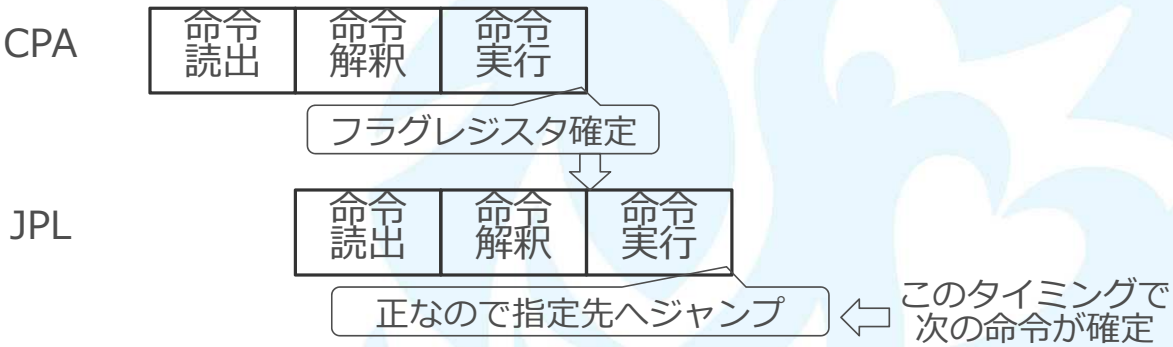
例：　条件分岐のとき



パイプラインのハザード

パイプラインが途切れて
うまく動かないときのこと

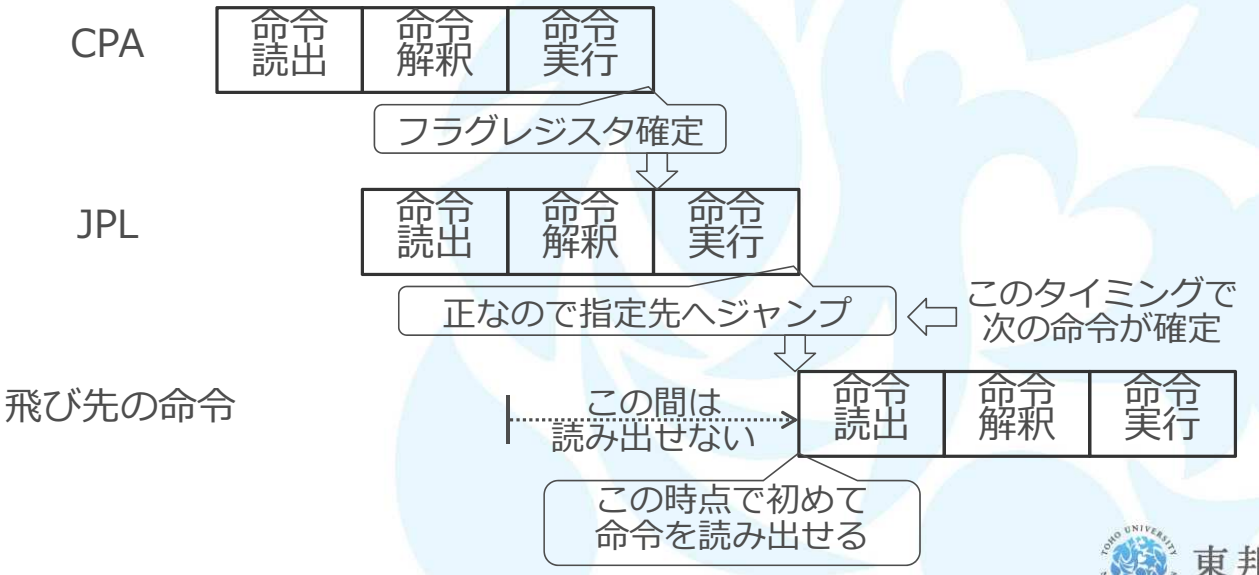
例：　条件分岐のとき



パイプラインのハザード

パイプラインが途切れて
うまく動かないときのこと

例：　条件分岐のとき



パイプラインのハザード

すべてうまく（同期して）動けば
計算どおりの性能が出るはずだが

パイプラインのハザード

すべてうまく（同期して）動けば
　計算どおりの性能が出るはずだが

この例のように、進めなくなることがあって
　その時は、期待した性能が出せない

ステージ数Ｓ倍になるはず(前回)

パイプラインのハザード

すべてうまく（同期して）動けば
　計算どおりの性能が出るはずだが

この例のように、進めなくなることがあって
　その時は、期待した性能が出せない

ステージ数Ｓ倍になるはず(前回)

パイプラインのハザードと呼ぶ

ハザードの分類

構造ハザード

メモリやレジスタなどの同時に使えない資源に同時にアクセスしようとしたとき

データハザード

次の命令へのデータの供給が間に合わないとき

制御ハザード

条件分岐で分岐が起こる場合に、次の命令が確定しないとき

ハザードの分類

構造ハザード

メモリやレジスタなどの同時に使えない資源に同時にアクセスしようとしたとき

データハザード

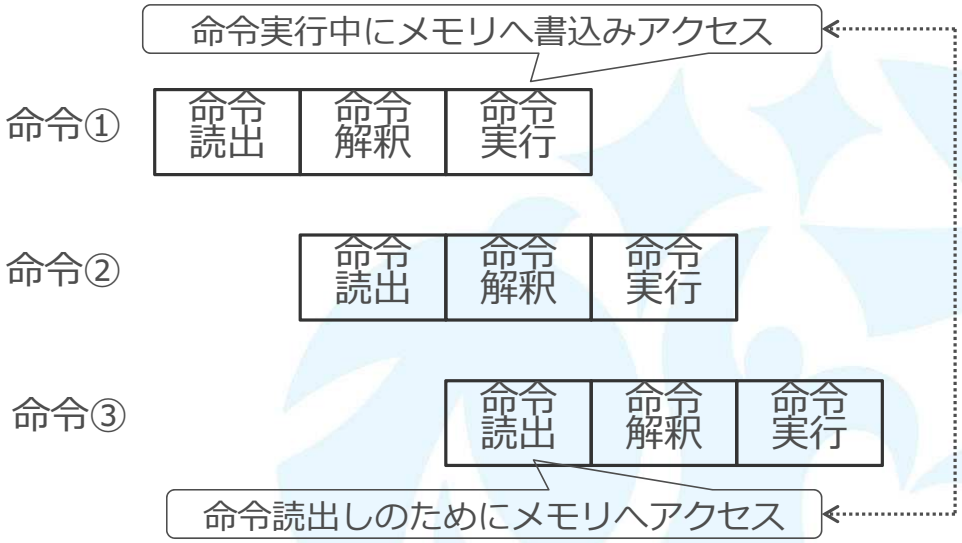
次の命令へのデータの供給が間に合わないとき

制御ハザード

条件分岐で分岐が起こる場合に、次の命令が確定しないとき

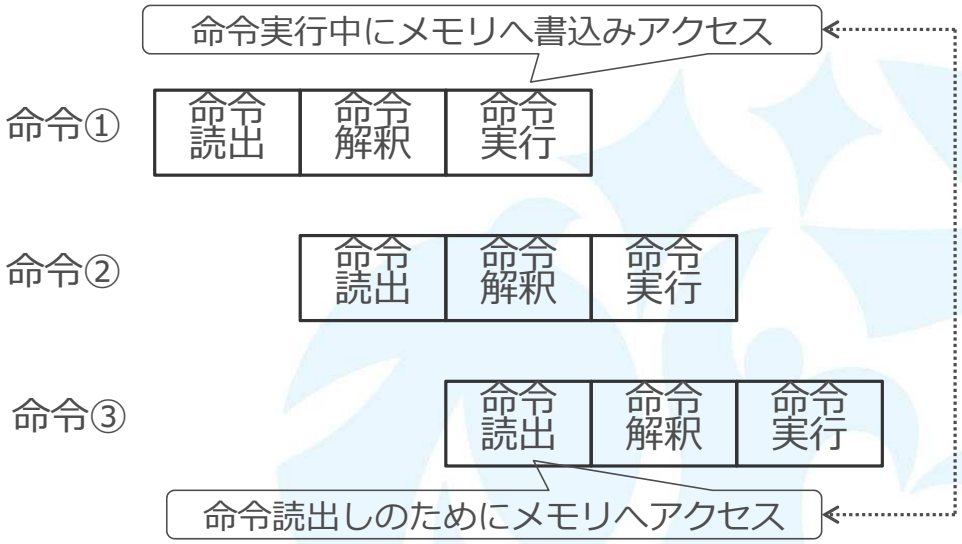
構造ハザード

例



構造ハザード

例



もし同じメモリハードウェアユニットへの
アクセスで、同時アクセスできない場合は
命令③を１つ後ろへずらさなければならない

ハザードの分類

構造ハザード

メモリやレジスタなどの同時に使えない資源に同時にアクセスしようとしたとき

データハザード

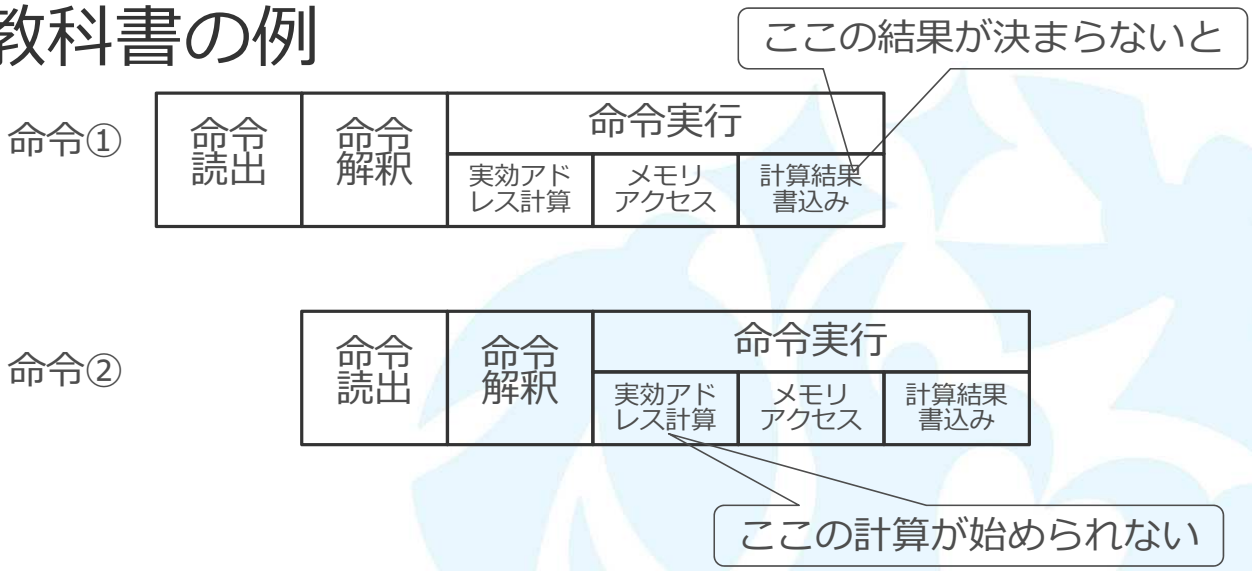
次の命令へのデータの供給が間に合わないとき

制御ハザード

条件分岐で分岐が起こる場合に、次の命令が確定しないとき

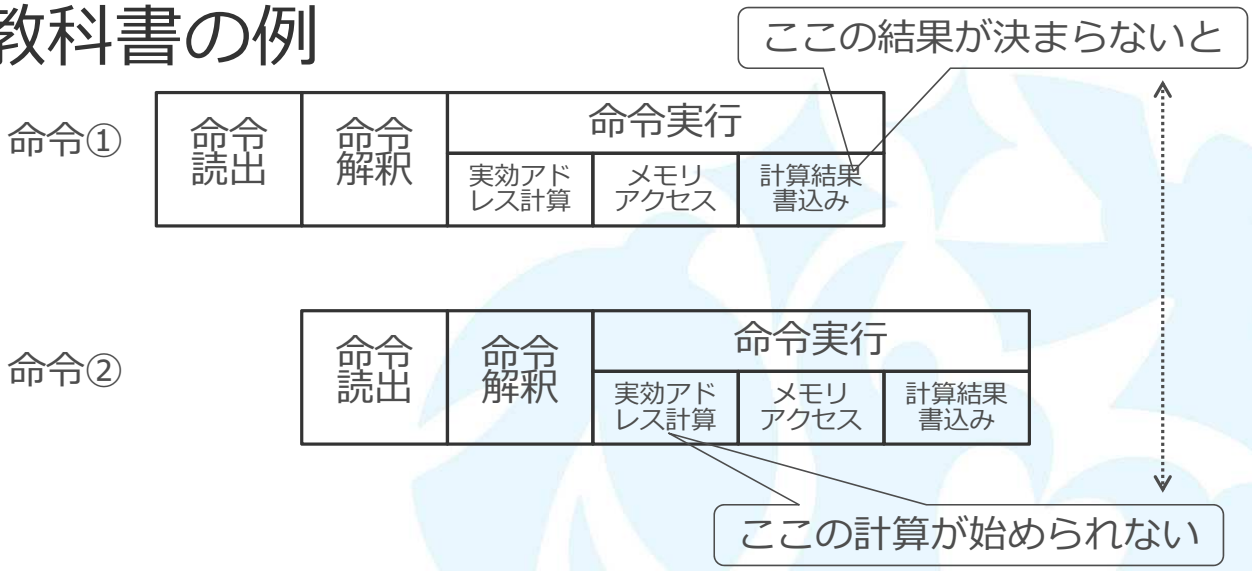
データハザード

教科書の例



データハザード

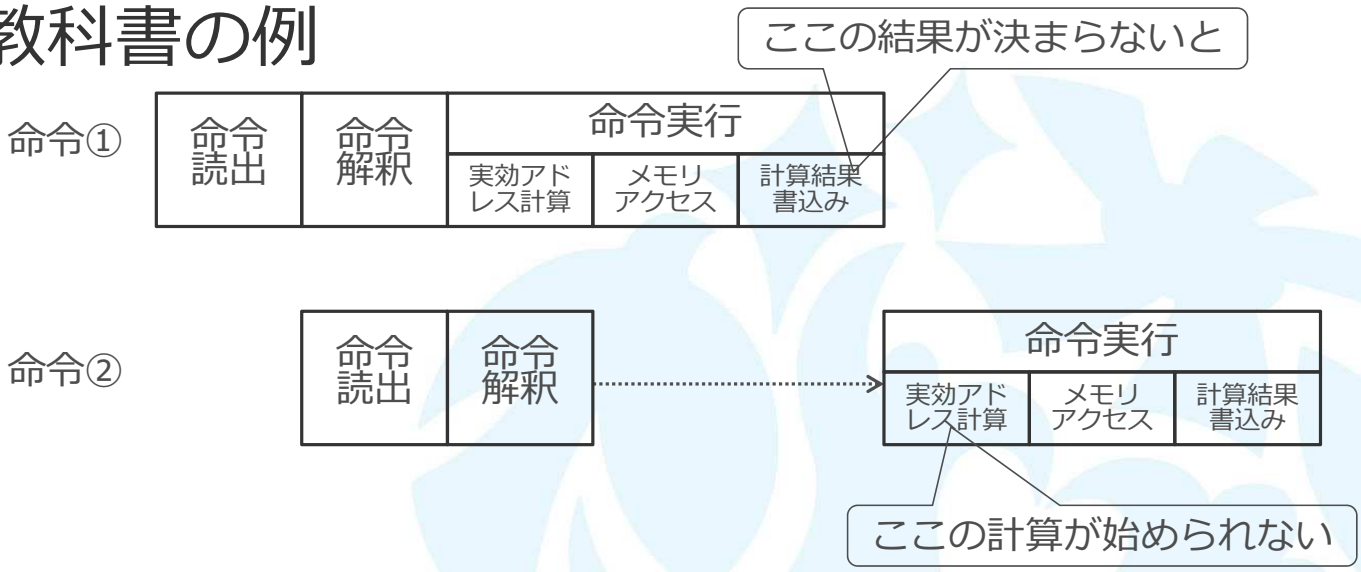
教科書の例



命令①で計算してレジスタGR5に書き込むとする
命令②は指標レジスタにGR5を指定していた場合、
①でGR5が確定しないと、②の実効アドレスの
計算が始められない

データハザード

教科書の例



命令②を２スロット分遅らせなければならない

RAW (Read After Write) の例
（書いてからでないと読めない）

データハザード

３つのパターン

RAW (Read After Write)
　データを書き込んでからでないと、読み出せない
WAR (Write After Read)
　読み出してからでないと、上書きできない
WAW (Write After Write)
　２つの命令が同じところに書き込む
　（元の命令の順序どおりに書き込まなければならない）

ハザードの分類

構造ハザード

メモリやレジスタなどの同時に使えない資源に同時にアクセスしようとしたとき

データハザード

次の命令へのデータの供給が間に合わないとき

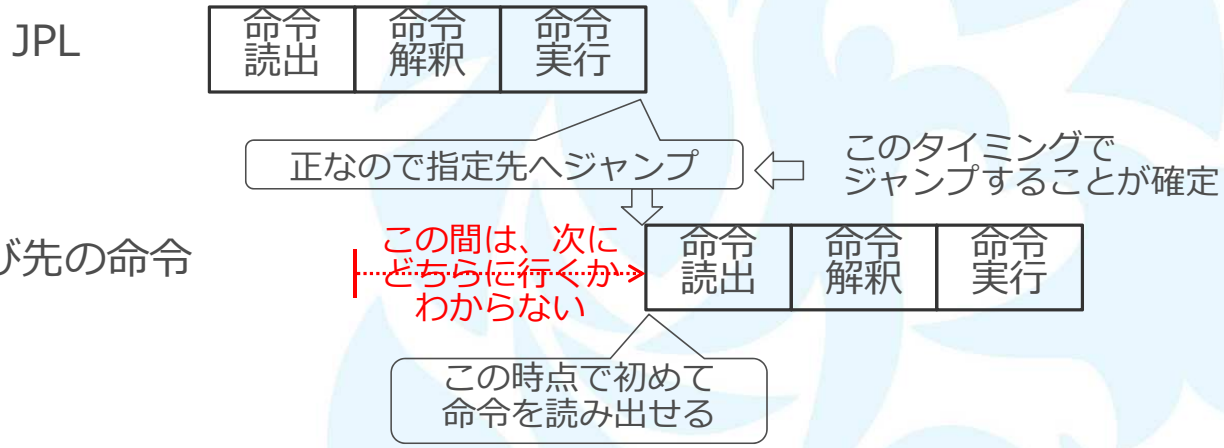
制御ハザード

条件分岐で分岐が起こる場合に、次の命令が確定しないとき

制御ハザード

条件分岐で、分岐先が決まるまで
命令読出しができない

フラグレジスタが正になっているとする



ハザードの分類

構造ハザード
メモリやレジスタなどの同時に使えない資源に同時にアクセスしようとしたとき

データハザード
次の命令へのデータの供給が間に合わないとき

制御ハザード
条件分岐で分岐が起こる場合に、次の命令が確定しないとき

制御ハザードの対策

遅延分岐

条件分岐命令の前にある無関係な命令を、
分岐の後に移動する

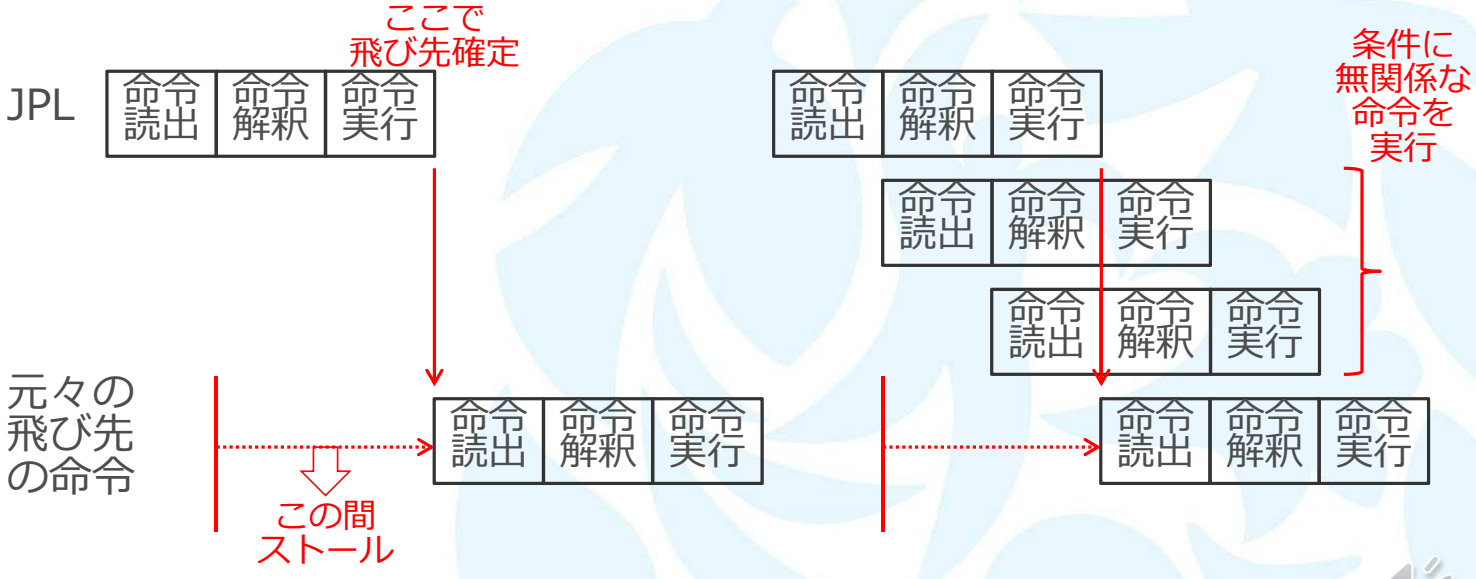
分岐予測

どちらに分岐するかを予測して次の命令を選ぶ
過去の分岐を記憶しておき、それに従う

制御ハザードの対策

遅延分岐

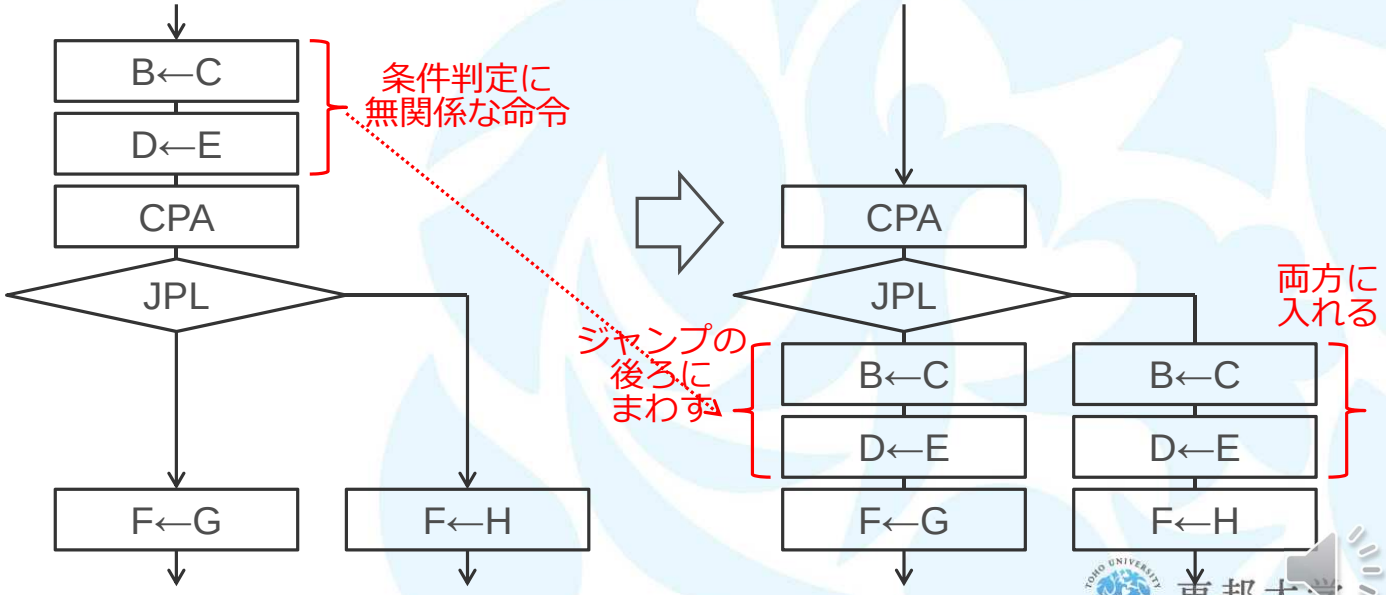
条件分岐命令の前にある無関係な命令を、
分岐の後に移動する



制御ハザードの対策

遅延分岐

条件分岐命令の前にある無関係な命令を、
分岐の後に移動する



制御ハザードの対策

分岐予測

どちらに分岐するかを予測して次の命令を選ぶ

その条件分岐命令での過去の分岐 (例: 1 回前)
の飛び先を記憶しておく
今回もそれと同じに飛ぶと予想して
その飛び先の命令列を読み始める
外れたらパイプラインをやり直す

制御ハザードの対策

遅延分岐

条件分岐命令の前にある無関係な命令を、
分岐の後に移動する

分岐予測

どちらに分岐するかを予測して次の命令を選ぶ
過去の分岐を記憶しておき、それに従う

さらなる高速化の技術

（教科書10.3節）

スーパーパイプライン

パイプラインのステージを多く（深く）する
段数Sのとき、理想的な性能はS倍のはず
Pentium IVで20ステージ

スーパースカラ

命令パイプラインを複数持ち、複数の命令を
同時にパイプラインで処理する
⇒命令の順序は保たれない可能性がある

さらなる高速化の技術

VLIW (very long instruction word)

1つの命令の中に複数の処理指示を埋め込む
異なるハードウェアユニットに対する指示を
同時に出すことによって、同時に処理
転送・浮動小数加算・同乗算・整数演算・分岐
ハードをどう使うかは、コンパイラが決めて
それを埋め込んだ命令を生成する

ベクトルコンピュータ

ベクトルデータに対して、演算パイプラインで
高速処理する

さらなる高速化の技術

マルチプロセッサ

複数の (独立した) CPUを並べて並列処理する
基本的に、プログラムが独立して走る

密結合 vs 疎結合
密結合：メインメモリを共有する
CPUは同じソフト・同じOSを共有
CPU数に上限（メモリを共有する仕組が限界）
「マルチコア」　Core 7i等で4～8コア程度
疎結合：メモリを共有せず、ネットワークで結合
共有データは転送しなければならない
規模に上限がない　（京は88,128 CPU）

パイプラインのハザードについて分かりましたか？

○

次へ

×