Task 2: Quantitative Problem Solving (E2)

Problem:
Several researchers have suggested that adding a register-memory addressing mode to a load-store machine might be useful. The idea is to replace sequences of:

```
LOAD Rx,0(Rb)
ADD Ry,Ry,Rx
```

by

```
ADD Ry,0(Rb)
```

Assume this new instruction will cause the clock period of the CPU to increase by 5%. Use the instruction frequencies for the gcc benchmark on the load-store machine from Table 1. The new instruction affects only the clock cycle and not the CPI.

1. What percentage of the loads must be eliminated for the machine with the new instruction to have at least the same performance?
2. Show a situation in a multiple instruction sequence where a load of a register (say Rx) followed immediately by a use of the same register (Rx) in an ADD instruction, could not be replaced by a single ADD instruction of the form proposed.

Solution:

Part 1: Percentage of Loads to be Eliminated

To determine the percentage of loads that must be eliminated for the machine with the new instruction to have at least the same performance, we need to compare the execution times before and after the introduction of the new instruction.

Given:

- The clock period of the CPU increases by 5% due to the new instruction.
- Instruction frequencies from Table 1:
  - LOAD: 22.8%
  - STORE: 14.3%
  - ADD: 14.6%
  - Other instructions: 43.4%

Let's denote:

- f_{load} as the fraction of LOAD instructions.
- f_{add} as the fraction of ADD instructions.
- f_{other} as the fraction of other instructions.

- $\text{Clock Rate}_{\text{old}}$ as the original clock rate.
- $\text{Clock Rate}_{\text{new}}$ as the new clock rate after the introduction of the new instruction.

From the problem:

- $f_{\text{load}} = 0.228$
- $f_{\text{add}} = 0.146$
- $f_{\text{other}} = 0.434$
- $\text{Clock Rate}_{\text{new}} = 0.95 \times \text{Clock Rate}_{\text{old}}$

The instruction count for the old version is:

$$\text{Instruction Count}_{\text{old}} = f_{\text{load}} \times I + f_{\text{add}} \times I + f_{\text{other}} \times I = I$$

The instruction count for the new version is:

$$\text{Instruction Count}_{\text{new}} = (f_{\text{load}} - x) \times I + f_{\text{add}} \times I + f_{\text{other}} \times I$$

where x is the fraction of LOAD instructions that are eliminated.

The clock cycle time for the old version is:

$$\text{Clock Cycle Time}_{\text{old}} = \frac{1}{\text{Clock Rate}_{\text{old}}}$$

The clock cycle time for the new version is:

$$\text{Clock Cycle Time}_{\text{new}} = \frac{1}{\text{Clock Rate}_{\text{new}}} = \frac{1}{0.95 \times \text{Clock Rate}_{\text{old}}}$$

We want the new version to have at least the same performance as the old version:

$$\text{CPU Time}_{\text{new}} \leq \text{CPU Time}_{\text{old}}$$

$$\text{Instruction Count}_{\text{new}} \times \text{Clock Cycle Time}_{\text{new}} \leq \text{Instruction Count}_{\text{old}} \times \text{Clock Cycle Time}_{\text{old}}$$

Substituting the values:

$$(I - xI) \times \frac{1}{0.95 \times \text{Clock Rate}_{\text{old}}} \leq I \times \frac{1}{\text{Clock Rate}_{\text{old}}}$$

Simplifying:

$$(1 - x) \times \frac{1}{0.95} \leq 1$$

$$1 - x \leq 0.95$$

$$x \geq 0.05$$

So, at least 5% of the LOAD instructions must be eliminated for the machine with the new instruction to have at least the same performance.

Part 2: Situation Where Replacement is Not Possible

Consider the following sequence of instructions:

```
LOAD Rx,0(Rb)
ADD Ry,Ry,Rx
LOAD Rz,0(Rb)
ADD Rz,Rz,Rx
```

In this sequence, the first LOAD and ADD can be replaced by:

```
ADD Ry,0(Rb)
```

However, the second LOAD and ADD cannot be replaced by:

```
ADD Rz,0(Rb)
```

because the value of `Rx` is used in the second ADD instruction, and it is not available in the memory location `0(Rb)`. Therefore, the sequence cannot be fully replaced by the new ADD instruction.

Conclusion:

1. At least 5% of the LOAD instructions must be eliminated for the machine with the new instruction to have at least the same performance.
2. A situation where a LOAD followed by an ADD cannot be replaced by the new ADD instruction is when the value loaded is used in a subsequent ADD instruction that cannot be replaced by the new instruction form.