8.6
Why Turing Machine is a "real" computer?

Why Turing Machine is a "real" computer?

TM can compute anything that a real computer can, if very very very tediously.

1. Add/multiply two numbers in binary representation.
2. Move input tape one position to the right.
3. Simulate a TM with two tapes.
4. Simulate a TM with many tapes.
5. Stack.
6. Subroutines.
7. Compile say any C program into a TM.
8. Conclusion: TM can do what a regular program can do.
9. Turing brilliant observation: A TM can simulate/modify another TM.
10. Modern equivalent: An interpreter can run a program that might be the interpreter itself (you don't say).

Why Turing Machine is a "real" computer?

TM can compute anything that a real computer can, if very very very tediously.

1. Add/multiply two numbers in binary representation.
2. Move input tape one position to the right.
3. Simulate a TM with two tapes.
4. Simulate a TM with many tapes.
5. Stack.
6. Subroutines.
7. Compile say any C program into a TM.
8. Conclusion: TM can do what a regular program can do.
9. Turing brilliant observation: A TM can simulate/modify another TM.
10. Modern equivalent: An interpreter can run a program that might be the interpreter itself (you don't say).

Why Turing Machine is a "real" computer?

TM can compute anything that a real computer can, if very very very tediously.

1. Add/multiply two numbers in binary representation.
2. Move input tape one position to the right.
3. Simulate a TM with two tapes.
4. Simulate a TM with many tapes.
5. Stack.
6. Subroutines.
7. Compile say any C program into a TM.
8. Conclusion: TM can do what a regular program can do.
9. Turing brilliant observation: A TM can simulate/modify another TM.
10. Modern equivalent: An interpreter can run a program that might be the interpreter itself (you don't say).

Why Turing Machine is a “real” computer?

TM can compute anything that a real computer can, if very very very tediously.

1. Add/multiply two numbers in binary representation.
2. Move input tape one position to the right.
3. Simulate a TM with two tapes.
4. Simulate a TM with many tapes.
5. Stack.
6. Subroutines.
7. Compile say any C program into a TM.
8. Conclusion: TM can do what a regular program can do.
9. Turing brilliant observation: A TM can simulate/modify another TM.
10. Modern equivalent: An interpreter can run a program that might be the interpreter itself (you don’t say).

Why Turing Machine is a “real” computer?

TM can compute anything that a real computer can, if very very very tediously.

1. Add/multiply two numbers in binary representation.
2. Move input tape one position to the right.
3. Simulate a TM with two tapes.
4. Simulate a TM with many tapes.
5. Stack.
6. Subroutines.
7. Compile say any C program into a TM.
8. Conclusion: TM can do what a regular program can do.
9. Turing brilliant observation: A TM can simulate/modify another TM.
10. Modern equivalent: An interpreter can run a program that might be the interpreter itself (you don’t say).

So what Turing Machines are good for?

1. Simplest mathematical way to describe a computer/program.
2. A good sandbox to argue about what programs can and can not do.
3. A terrible counter-intuitive model, completely unlike real world programs.
4. TM = PROGRAM.

So what Turing Machines are good for?

1. Simplest mathematical way to describe a computer/program.
2. A good sandbox to argue about what programs can and can not do.
3. A terrible counter-intuitive model, completely unlike real world programs.
4. TM = PROGRAM.

So what Turing Machines are good for?

1. Simplest mathematical way to describe a computer/program.
2. A good sandbox to argue about what programs can and can not do.
3. A terrible counter-intuitive model, completely unlike real world programs.
4. TM $=$ PROGRAM.

Universal Turing Machine

Turing Machine that simulates another Turing Machine

UTM: A Turing machine that can simulate another Turing machine.

1. Programs can self replicate.
2. Program can modify themselves (a big no no nowadays).
3. Program can rewrite a program.
4. Turing had created a Pandora box...
   ...which we will open in the next lecture.

THE END

...

(for now)