

Unit B6 Supplement, 3/4/05

Ex B5 Compiled Heuristics

Here is a (edited and expanded) collection of ideas from your Ex B5, classified according to the involved techniques. In some cases, wording is subtle.

TM Construction [Useful for simple problems, e.g., combinatory problems]

1. If it is possible to create an algorithm (a terminating TM) to solve the problem, it is decidable.
2. If it is possible to create a TM to “recognize” the problem/language (but not guaranteed to terminate), it is TM-recognizable.

Note: The following cases are insufficient as conditions.

- Not being able to come up with an algorithm/TM (in the future?)
- Potential of looping (too informal; why did we discuss the main problems in detail?)
- Solving at least one input (a single input is not enough)

Set operation [Useful when multiple properties are known]

3. If the problem is TM-recognizable and undecidable, it is semi-decidable.
4. If the problem is both TM-recognizable and co-TM-recognizable, it is decidable.
5. If the complement of the problem is decidable, it is decidable.
6. If the complement of the problem is semi-decidable, it is non-TM-recognizable.
7. If the complement of the problem is non-TM-recognizable, it is undecidable. [added]

Reduction [Useful when another problem is known]

8. If the problem can be reduced to a known decidable (TM-recognizable) problem, the former problem is decidable (TM-recognizable).
9. If a known undecidable (non-TM-recognizable) problem can be reduced to the problem, the latter problem is undecidable (non-TM-recognizable).
10. If the *computability analysis* of the problem can be reduced to that of a known decidable (TM-recognizable) problem, the former problem is decidable (TM-recognizable). [added]
11. If the *computability analysis* of a known undecidable (non-TM-recognizable) problem can be reduced to that of the problem, the latter problem is undecidable (non-TM-recognizable). [added]

Countability [Useful when there are more languages (uncountable) than all of TMs (countable)]

12. If the problem is about a non-trivial property of a TM/language, it is undecidable (Rice’s theorem).
13. If the problem is a non-empty, proper subset of an uncountable set (or the power set of a countably infinite set), it is undecidable. [added]

Note: The following cases are insufficient as conditions.

- Uncountable set (there are trivial decidable problems, i.e., constant problems)
- Countably infinite set (cf. L_d is countable)
- Finite set (cf. a truly random problem)

Other [Useful for complex representations, e.g., algorithm, function, logic, grammar]

14. If the problem involves a mechanism that allows counting (natural numbers) and self-reference, it is (probably) undecidable. [I am not prepared to do a rigorous justification.]

Note: The following cases are insufficient as conditions.

- Universality or self-reference (this could happen in a limited domain which cannot count)

// End