If a set B is m-reducible to a decidable set A, $B \leq_m A$, then B is also decidable. Suppose A is undecidable, but we have access to an oracle for A that answers questions about the membership of numbers in A. In which case, we can use it to answer questions about B. Here's a general scheme:

(a) Only one question is asked.

(b) The answer is taken as the answer to the original question.

Here are a couple of instances that are related but don't quite follow the rules above:

(a) Given an oracle for the set A can we answer questions about A^c? Yes, but (b) in the scheme above is being reversed.

(b) Given an oracle for A can we answer questions about $A \times A$? Yes, but it seems like (a) in the scheme above isn't quite right as we're asking two questions.

We now seek a more general notion of reducibility:

Definition 1. (a) Let us say that B is Turing Reducible to A if there exists an algorithm that decides the set B using an oracle answering questions about A. We denote this by $B \leq_T A$.

(b) If B is reducible to A, we also say that B is A-decidable.

Theorem 1. *(a) If $B \leq_m A$, then $B \leq_T A$.*

(b) $A \leq_T \mathbb{N}_0 \setminus A$ for any A.

(c) If $A \leq_T B$ and $B \leq_T C$ then $A \leq_T C$.

(d) If $A \leq_T B$ and B is decidable, then A is decidable.

Remark 1. Notice that (b) implies that it's possible that a nonenumerable set can be Turing Reducible to an enumerable set, which is impossible for m-reducibility.

We can define the notion of an A-computable funciton similarly to how e defined what it meant to be A-decidable.

Definition 2. A funcction f is said to be A-computable if there exists an algorithm M, with calls to an oracle, that computes f if the calls are correctly answered by the A-oracle.

- If f is defined at x, then $f(x)$ is returned and the algorithm halts.
- If $x \notin \operatorname{Dom} f$, then it does not halt.

Theorem 2. *A partial function f is computable relative to a total function α if and only if it is computable relative to the graph of the function α.*

We now work through a whole slew of definitions:

Definition 3.

(a) A function with natural values defined on a finite set of $\mathbb{N}_0$ is called a pattern. Any patter is defined by a set of pairs (arg , value).

(b) Two patterns are called coherent if the union of their graphs is the graph of a function. There is no place where both are defined and take on different values.

(c) Let (x, y, t) be a triple where $x, y \in \mathbb{N}_0$ and t is a pattern. We will say that two triples (x_1, y_1, t_1) and (x_2, y_2, t_2) are incompatible if the patterns are coherent if $x_1 = x_2$ but $y_1 \neq y_2$.

(d) A set M will be said to be consistent if it contains no incompatible triples.

We now have the following construction:

Let M be a consistent set and α some function. Consider triples $(x, y, t) \in M$ such that $\Gamma(t) \subset \Gamma(\alpha)$. All patters are coherent so M is consistent. No two of the triples can have equal first and different second components. If we omit the 3^{rd} element of the triples, we get the graph of a function (which is probably partial). This function will be denoted $M[\alpha]$.

Theorem 3. *A partial function $\mathbb{N}_0 \to \mathbb{N}_0$ is computable relative to a total function $\alpha : \mathbb{N}_0 \to \mathbb{N}_0$ if and only if there exists an enumerable consistent set of triples M such that $f = M[\alpha]$.*

Relativization

Let us fix a total function α the entire theory of computable functions can be relativized with respect to α. So the theory we've developed can be applied to α-computable functions. The proofs are the same with small changes. In particular, we can defined the notion of a set enuemrable with respect to α or α-enumerable:

(a) As the domain of an α-computable function.

(b) As the range of an α-computable function.

(c) As the projection of an α-decidable set.

Let E be an arbitrary set of pairs (x, t) where x is a numer and t is a pattern. We take a total function α and pick out from the set E the pairs whose 2^{nd} components are parts of α. The first components will be denoted by $E[\alpha]$.

Theorem 4. *A set X is α-enumerable if and only if $X = E[\alpha]$ for some enumerable set E.*

Proof. Let X be the domain of a function f that is α-computable. Let $f = M[\alpha]$ for some enumerable set M that is consistent. Let us delete the 3^{rd} component from each triple in M. We obtain an enumerable set of pairs, call it E. Then $E[\alpha]$ is the domain of f so $E[\alpha] = X$.

Conversely, suppose that $X = E[\alpha]$ for some function α. Now consider the set M obtained by inserting 0 between the components for each pair of E. The set M is consistent and $M[\alpha] \equiv 0$ defined on $X = E[\alpha]$. □

Theorem 5. *Let α be a total function. There exists a binary computable function universal for the class of unary α-computable functions.*