2.3 Intractability; Compression and Speed-up Theorems

The t-***restriction*** u_t of u aborts and outputs 1 if $u(x)$ does not halt within $t(x)$ steps, i.e. u_t computes the t-***Bounded Halting Problem*** ***(t-BHP)***. It remains complete for the closed under negation class of functions computable in $o(t(x))$ steps. ($O(\|p\|^2)$ overhead is absorbed by $o(1)$ and padding p.) So, u_t is not in the class, i.e. cannot be computed in time $o(t(x))$ [Tseitin 56]. (And neither can be any function agreeing with t-BHP on a ***dense*** (i.e. having strings with each prefix) subset.) E.g. $2^{\|x\|}$-BHP requires exponential time.

However for some trivial input programs the BHT can obviously be answered by a fast algorithm. The following theorem provides another function $P_f(x)$ (which can be made a predicate) for which there is only a finite number of such trivial inputs. We state the theorem for the volume of computation of Multi-Head Turing Machine. It can be reformulated in terms of time of Pointer Machine and space (or, with smaller accuracy, time) of regular Turing Machine.

Definition: A function $f(x)$ is ***constructible*** if it can be computed in volume $V(x) = O(f(x))$.

Here are two examples: $2^{\|x\|}$ is constructible, as $V(x) = O(\|x\| \log \|x\|) \ll 2^{\|x\|}$.
Yet, $2^{\|x\|} + h(x)$, where $h(x)$ is 0 or 1, depending on whether $U(x)$ halts within $3^{\|x\|}$ steps, is not.

Compression Theorem [Rabin 59]. For any constructible function f, there exists a function P_f such that for all functions t, the following two statements are equivalent:

1. There exists an algorithm A such that $A(x)$ computes $P_f(x)$ in volume $t(x)$ for all inputs x.
2. t is constructible *and* $f(x) = O(t(x))$.

Proof. Let t-***bounded Kolmogorov Complexity*** $K_t(i|x)$ of i given x be the length of the shortest program p for the Universal Multi-Head Turing Machine transforming x into i with $< t$ volume of computation. Let $P_f(x)$ be the smallest i, with $2K_t(i|x) > \log(f(x)|t)$ for all t. P_f is computed in volume f by generating all i of low complexity, sorting them and taking the first missing. It satisfies the Theorem, since computing $i{=}P_f(x)$ faster would violate the complexity bound defining it. (Some extra efforts can make P Boolean.) $\square$

Speed-up Theorem [Blum 67]. There exists a total computable predicate P such that for any algorithm computing $P(x)$ in volume $t(x)$, there exists another algorithm doing it in volume $O(\log t(x))$.

Though stated here for exponential speed-up, this theorem remains true with log replaced by any computable unbounded monotone function. In other words, there is no even nearly optimal algorithm to compute P.

The general case. So, the complexity of some predicates P cannot be characterized by a single constructible function f, as in Compression Theorem. However, the Compression Theorem remains true (with harder proof) if the requirement that f is constructible is dropped (replaced with being computable).[4]
In this form it is general enough so that every computable predicate (or function) P satisfies the statement of the theorem with an appropriate computable function f. There is no contradiction with Blum's Speed-up, since the complexity f (not constructible itself) cannot be reached. See a review in [Seiferas, Meyer 95].

[4]The proof stands if constructibility of f is weakened to being ***semi-constructible***, i.e. one with an algorithm $A(n,x)$ running in volume $O(n)$ and such that $A(n,x){=}f(x)$ if $n{>}f(x)$. The sets of programs t whose volumes (where finite) satisfy either (1) or (2) of the Theorem (for computable P,f) are in Σ^0_2 (i.e. defined with 2 quantifiers). Both generate monotone classes of constructible functions closed under $\min(t_1,t_2)/2$. Then any such class is shown to be the $\Omega(f)$ for some **semi-constructible** f.