# Math 29: Low and High Sets

May 23rd, 2022

We're interested in knowing exactly how far the jump can go. Of course, technically it goes up forever because we can iterate it over and over again. However, how weak can the jump of a set be? How powerful can a single jump be?

## 1 Low Sets

A set $X$ is **low** if $\emptyset' \equiv_T X'$. In other words, the jump is as small as it could possibly be. Note that this implies $X <_T \emptyset'$, as $X <_T X' = \emptyset'$. Furthermore, it suffices to prove $X' \leq_T \emptyset'$, as $\emptyset \leq_T X$ implies $\emptyset' \leq_1 X'$ for all $X$ by previous results.

Note that we haven't explicitly excluded the computable sets. Whether or not to include them is much like the opposite of whether or not to include 1 in the primes: not including them still gives us a perfectly reasonable definition, but requires us to rephrase many key results or theorems to say "computable or low" rather than simply "low."

Regardless, we need to justify that low is not an alias for computable. In fact, we can show that there is a low simple set, answering two questions with one construction: it will be a noncomputable low set because it is simple, and it will be a simple set which is not complete.

We modify the construction of a simple set to produce a simple set which is low. The basic idea is that, given an enumeration of a c.e. set $A_s$, we can define the computable function

$$g(e,s) = \begin{cases} 1 & \Phi_{e,s}^{A_s}(e)\downarrow \\ 0 & \text{otherwise} \end{cases}$$

Clearly if the limit along $s$ exists for all $e$, then the limit of $g$ will be the characteristic function of $A'$, which will be jump computable by the limit lemma. However, the danger could be that $\Phi_{e,s}^{A_s}$ keeps converging and then later diverging infinitely often: Imagine the machine which reads oracle values until it finds an $n$ such that at least half of the $oracle(k)$ calls belows $n$ are 0. This may halt early, but then diverge if we keep adding elements. So without being judicious, we could see convergence infinitely often, but continously break each one so that it does not converge in the limit.

To fix this, we will add a restraint function to the traditional simple set construction as inspired by Friedberg-Muchnik to constrain the use and ensure this does not happen. Once we see a computation $\Phi_{e,s}^{A_s}(e)$ converge, then we set the restraint so that lower priority requirements don't break this computation. Then each computation will break at most finitely often, as there will only be finitely many higher priority requirements, thereby ensuring that this strategy will work.

This is far from the only noncomputable low set. In fact, there are many low sets, and we have a very reliable way to find low sets with desirable properties. Recall that a binary tree $T$ is a set of finite binary strings which is closed under predecessors, i.e. if $\tau$ is a longer binary string than $\sigma$ which agrees with $\sigma$ on its domain, then $\tau \in T$ implies $\sigma \in T$.

A **class** is a subset of $2^\omega$, i.e. a set of sets of natural numbers. A class is a $\Pi_1^0$ class if it can be written as exactly the paths through some computable tree $T$.

**Theorem 1.** *(Low Basis Theorem) Every nonempty $\Pi_1^0$ class contains a low element.*

**Proof:** Notice that $\emptyset'$, while it cannot in general solve $Fin$, it can determine whether or not a tree is finite: a tree is finite if and only if there is some $n$ such that $T \cap 2^n = \emptyset$, i.e. there is some highest level. This is a $\Sigma_1^0$ question, as it is asking if an $n$ exists such that $\forall \tau \in 2^n$ (a bounded quantifier), $\tau \notin T$. Therefore, $\emptyset'$ can uniformly determine if a computable tree is finite.

Therefore, given any infinite computable tree $T$, define $T_0 = T$ and $T_{e+1}$ via: the set

$$U_e = \{\sigma : \Phi_{e,|\sigma|}^{\sigma}(e) \uparrow\}$$

is a computable tree, and thus so is $T_e \cap U_e$. If $T_e \cap U_e$ is infinite, define $T_{e+1} = T_e \cap U_e$. Otherwise, let $T_{e+1} = T_e$. $\emptyset'$ can deteremine which case to follow at each stage, and thus can compute this sequence of trees.

Let $\hat{T} = \bigcap_{e\in\omega} T_e$. Then $\hat{T}$ is infinite by compactness of $2^\omega$, and thus contains an element $X$ by Weak König's Lemma.

**Proof:** (Cont.) But notice that if $T_{e+1} = T_e \cap U_e$, it must be the case that $\Phi_e^X(e)\uparrow$. If not, then $U_e \cap T_e$ is finite, and therefore $\Phi_e^Y(e)\downarrow$ for every $Y \in T_e$, so in particular, $\Phi_e^X(e)\downarrow$. Therefore, $\emptyset'$ can compute $X'$ by checking which case we used at each step.

In particular, there are low separating sets for any two computably inseparable c.e. sets, and complete, consistent extensions of Peano Arithmetic which are of low degree. These follow since we showed that each can be represented as the paths through a computable tree.

It turns out that there are plenty of low sets, and plenty of low c.e. sets.

**Theorem 2.** *(Sacks' Splitting Theorem) For any c.e. set $A$, there are disjoint c.e. sets $B$ and $C$ such that $B \sqcup C = A$ which are both low.*

In particular, there are two low sets which can compute the halting problem when working together.

### 1.1 Low $n$

A set $X$ if $\mathrm{Low}_n$ if $\emptyset^{(n)} \equiv_T X^{(n)}$. That is, its $n$-th jump is as small as it could possibly be. Notice that a $\mathrm{Low}_k$ set is $\mathrm{Low}_n$ whenever $k < n$.

## 2 High Sets

Conversely, a set $X$ is **high** if $X \leq_T \emptyset'$ and $\emptyset'' \equiv_T X'$. In other words, $X$ is jump computable, and its jump is as big as it could possibly be. The halting problem is clearly high, as is any complete c.e. set. In particular, any set which computes a fixed point-free function is high by Arslanov's completeness criterion.

As in the case for lowness, we need to check that high is an interesting property, not just a different name for being complete. It turns out, they are not the same thing.

**Theorem 3.** *(Martin's High Domination Theorem) A set $X$ is high if and only if $A$ computes a function $f$ which dominates all total computable functions.*

**Proof:** Suppose that $X$ computes a function $f$ which dominates every total computable function. Then the function $g(e,s)$ defined via

$$g(e,s) = \begin{cases} 1 & \forall z \leq s \ \phi_{e,f(s)}(z)\downarrow \\ 0 & \text{otherwise} \end{cases}$$

**Proof:** (Cont.) For any $e$ which has $\varphi_e$ total, the function $\theta(x)$ defined to be the least $s$ such that $\varphi_{e,s}(x)\downarrow$ is defined, and thus $\theta$ is total computable. Therefore $f(s) \geq \theta(s)$ for all but finitely many $s$, so $g(e,s)$ will be 1 after the last $s$ where it is smaller. Therefore, the limit along all $e$ in $Tot$ will exist and equal 1. Conversely, for any $e$ not in $Tot$, $g(e,s)$ will be 0 for any $s$ larger than the least element of $W_e^c$, and will have limit 0.

This proves that $Tot$ is limit computable from $X$, which by the relativized limit lemma shows $Tot \leq_T X'$. Because we know $Tot$ computes $\emptyset''$, $X$ is therefore high.

Now suppose $X$ is high. Then because $Tot$ computes $\emptyset''$, it is computable from $X'$ and therefore limit computable in $X$ via some $g(e,s)$ by the relativized limit lemma. Define $t(e,s)$ to be the least number larger than $s$ such that $g(e,t)=0$ or, for every $x \leq s$, $\varphi_{e,t}(x)\downarrow$. In other words, $t(e,s)$ is the smallest time larger than $s$ by which $\varphi_e$ looks total up to $s$ by time $t$, or $g(e,t)$ is predicting that $\varphi_e$ is not total. Notice that such a $t$ always exists because $\varphi_e$ is total or $g(e,s)$ is eventually zero.

Define $f(s)$ to be the maximum of $t(e,s)$ over $e \leq s$. Then for all $e$ in $Tot$, $f(s) \geq \varphi_e(s)$ for all $s \geq t$. This is because, by convention the time at which a computation converges is larger than the output, and we are choosing $f$ to be the maximum over all times in a certain set which contains the times $\varphi_e(s)$ converge.

Thus $f$ dominates all total computable functions, and it is $X$ computable because $g$ is.

We can now construct an example of a noncomplete high set. Using the jump as an oracle, we build a function $f$ which dominates every total computable function but does not compute $\emptyset'$. We do so by choosing finite $\omega$-strings, i.e. finite strings of natural numbers, in a similar way as we have done for finite binary strings.

We'll set $\sigma_0 = \emptyset$. Given $\sigma_s$, define $\sigma_{s+1}$ as follows: Say that $e$ "looks total up to $n$" if there is some $t$ such that $\varphi_{e,t}(x)\downarrow$ for all $x \leq n$. Notice that $\emptyset'$ can determine if $e$ looks total up to $n$ because this is a $\Sigma^0_1$ question.

Now look for a $\tau$ properly extending $\sigma$ such that $\varphi_e(x) \leq \tau(x)$ for all $\sigma| < x \leq |\tau|$ and all $e \leq_s$ which look total up to $|\tau|$, and an $x$ such that $\Phi^\tau_s(x)\downarrow \neq K(x)$. If there is such a $\tau$ and $x$, let $\sigma_{2s+1}=\tau$. If not, let $\sigma_{2s+1}=\sigma_{2s}$. Similarly, this is a $\Sigma^0_1$ question, so $\emptyset'$ can determine if such a $\tau$ exists.

Let $f = \bigcup_s \sigma_s$. Notice that $f$ is infinite because whenever $\Phi_s$ does not use the oracle, it cannot compute $K(x)$, and therefore there must be $\tau$ which extends our current $\sigma$, because all $\tau$ needs to do is add one number larger than finitely many function values to $\sigma$.

Furthermore, $f$ dominates every total computable function. If $\varphi_k$ is total, then it looks total up to $n$ for all $n$, so for $s \geq k$, we are only extending by strings which dominate $\varphi_k$.

Lastly, $f$ does not compute $K$. If it did, i.e. $\Phi_s^f = K$, then it must do so via $\Phi_s$ for which there were no such $\tau$ and $x$, i.e. for all suitable $\tau$ and $x$, $\Phi_s^\tau(x)\downarrow$ implies $\Phi_s^\tau(x) = K(x)$. But then $K$ is computable: hard-code in the information of which $e \leq s$ are total, and let $m$ be such that all other $e \leq s$ have $\varphi_e$ diverge on some $x \leq m$. That is, we can computably determine for a fixed $s$ which $\tau$ are large enough to be possible extensions of $\sigma_s$. Then to compute $K$, hunt for a $\tau$ extending $\sigma_s$ whose length is at least $m$ which is larger than all total $\varphi_e$, $e \leq s$ and has $\Phi_s^\tau(x)\downarrow$. Then it must be the case that this converges to $K(x)$, and we'll eventually find such a $\tau$ because $\Phi_s^f = K$. This is a contradiction, so $f$ must not compute $K$.

## 2.1 High $n$

A set $X$ if $\mathrm{High}_n$ if $\emptyset^{(n+1)} \equiv_T X^{(n)}$. In other words, the $n$-th jump is as big as it could be assuming $X$ didn't start out above the jump. Notice that a $\mathrm{High}_k$ set is $\mathrm{High}_n$ whenever $n < k$.