Tracing and domination in the Turing degrees

George Barmpalias

Victoria University of Wellington

Heidelberg, July 2009

Plan of the talk

- Highness notions
- Questions by Nies and Simpson
- Tracing and domination
- General theorem (Turing degrees)
- Applications (Randomness)
- Further questions
- References

Computability and Randomness

- Randomness notions have enriched classical computability theory
- Lowness notions like K-triviality revealed new ideals in the Turing degrees
- Highness notions like almost everywhere domination were key in the study of the reverse mathematics of measure theory

Almost everywhere domination

- Given functions $f, g : \mathbb{N} \to \mathbb{N}$, we say that f dominates g if $f(n) \geq g(n)$ for almost all $n \in \mathbb{N}$.
- (Dobrinen and Simpson) A Turing degree $\mathbf{a}$ is called a.e. dominating, if for almost all $X \in 2^\omega$ and all functions $g \leq_T X$, there is a function $f \leq_T \mathbf{a}$ which dominates g.
- Kurtz showed that $\mathbf{0}'$ is a.e. dominating.

Highness and domination

- Martin showed that A is high iff it can compute a function which dominates all computable functions.
- Hence a.e. domination can be seen as a strengthening of highness
- The class of a.e. dominating degrees lies strictly in between high and complete degrees
- even in the local structure of computably enumerable degrees.

Highness and domination

- Martin showed that A is high iff it can compute a function which dominates all computable functions.
- Hence a.e. domination can be seen as a strengthening of highness
- The class of a.e. dominating degrees lies strictly in between high and complete degrees
- even in the local structure of computably enumerable degrees.

> *Another lowness/domination notion: A degree is called* **array computable** *if there is a function* $f \leq_{wtt} \emptyset'$ *which dominates all functions computable in it.*

Almost everywhere domination and degrees

- There has been an interest in clarifying the connection of a.e. domination with concepts from classical computability.
- What role does it play in the partial ordering of the Turing degrees?
- Can it be expressed purely in degree theoretic terms?

  …without resorting to measure or randomness?

Highness in the Turing degrees

In the Turing degrees, high degrees have special properties that distinguish them:

- every high Δ^0_2 degree bounds a minimal degree (Cooper)
- In the computably enumerable degrees, every high degree bounds a minimal pair (Cooper)

Questions

In this spirit, the following questions were asked:

- (Simpson, 2006) Is there a minimal a.e. dominating degree?
- (Nies, Problem 8.6.4 in his book) Is there a c.e. traceable a.e. dominating degree?

Questions

In this spirit, the following questions were asked:

- (Simpson, 2006) Is there a minimal a.e. dominating degree?
- (Nies, Problem 8.6.4 in his book) Is there a c.e. traceable a.e. dominating degree?

Contrast: There exist high minimal and c.e. traceable degrees.

Questions

In this spirit, the following questions were asked:

- (Simpson, 2006) Is there a minimal a.e. dominating degree?

- (Nies, Problem 8.6.4 in his book) Is there a c.e. traceable a.e. dominating degree?

Contrast: There exist high minimal and c.e. traceable degrees.

Our main result answers both of these questions in the negative.

Highness notions of interest (Greenberg, Miller, Simpson, Kjos-Hansen,...)

Turing complete $\underset{\not\Leftarrow}{\Rightarrow}$ a.e. dominating $\underset{\not\Leftarrow}{\Rightarrow}$ JT-complete

$\Downarrow \not\Uparrow$

$\emptyset'$-tracing

Traceability

- A c.e. trace with bound g is a uniformly c.e. sequence (U_i) such that $|U_i| < g(i)$.
- A function f has trace (U_i) if $f(i) \in U_i$ for all i.
- A is jump-traceable by B if every partial computable function relative to A can be traced by a B-c.e. trace with computable bound.
- B is A-tracing (or A is c.e. traceable by B) if every A-computable function can be traced by a B-c.e. trace with computable bound.

Diagram again

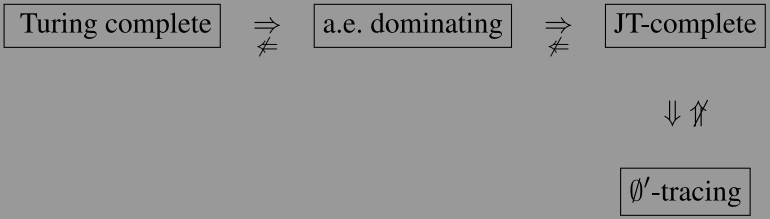


Diagram again

Turing complete $\underset{\not\Leftarrow}{\Rightarrow}$ a.e. dominating $\underset{\not\Leftarrow}{\Rightarrow}$ JT-complete

$\Downarrow \not\Uparrow$

$\emptyset'$-tracing

Question: Which of these notions imply highness?

The main result

Theorem

If **c** *is c.e. and is c.e. traceable by* **a***, then no function that is computable in* **c** *dominates every function computable in* **a***.*

The main result

Theorem
If **c** *is c.e. and is c.e. traceable by* **a***, then no function that is computable in* **c** *dominates every function computable in* **a***.*

Roughly: if **a** is ‘high’ in some sense over **c**, then it is not ‘low’ over **c** in another sense.

Special case of interest

Corollary

If $\mathbf{0}'$ *is c.e. traceable by* $\mathbf{a}$*, then no* $\mathbf{0}'$*-computable function dominates every function computable in* $\mathbf{a}$*.*

Special case of interest

Corollary

If $\mathbf{0}'$ *is c.e. traceable by* $\mathbf{a}$*, then no* $\mathbf{0}'$*-computable function dominates every function computable in* $\mathbf{a}$*.*

Recall: A degree $\mathbf{a}$ is called array computable if there is $f \leq_{wtt} \emptyset'$ which dominates all functions computable in $\mathbf{a}$.

Corollary

No almost everywhere dominating function has minimal degree

Corollary

No almost everywhere dominating function has minimal degree

Constrast: There are high minimal degrees (Cooper):

Corollary

No almost everywhere dominating function has minimal degree

Constrast: There are high minimal degrees (Cooper):

There are functions of minimal degree that dominate all computable functions.

Proof of Corollary

- Let $\mathbf{c} = \mathbf{0}'$ in the theorem.
- If $\mathbf{a}$ is a.e. dominating, then $\mathbf{0}'$ is c.e. traceable by $\mathbf{a}$.
- By main result, no $\mathbf{0}'$-computable function dominates all functions computable in $\mathbf{a}$.
- In particular, $\mathbf{a}$. is array non-computable.
- But all minimal degrees are array computable (Downey, Jockusch, Stob).

Diagram once again

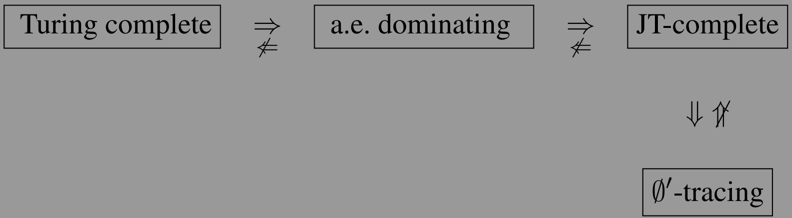
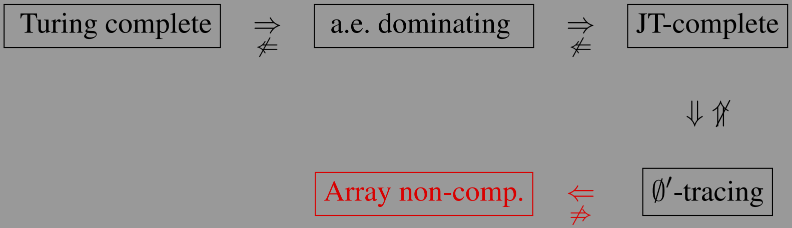


Diagram once again



Proof idea of Main result

In a computable construction, we build

- a C-computable function Θ^C
- a Turing functional Γ

such that

$$R_e : \text{for all } X, \text{ if } \Theta^C(e) \in V_e^X \quad \left\{ \begin{array}{ll} \text{(a)} & \forall n \in \mathbb{N}^{[e]},\ \Gamma^X(n)\downarrow \\ \text{(b)} & \exists n \in \mathbb{N}^{[e]},\ \Gamma^X(n) > \Phi^C(n) \end{array} \right\}$$

where Φ^C is a given C-computable function.

Main challenge

How to reduce a second order quantifier to the finite computation $\Theta^C(e)\downarrow$?

Main challenge

How to reduce a second order quantifier to the finite computation $\Theta^C(e)\downarrow$?

Answer: use the compactness of the Cantor space.

Further questions

- Can a.e. domination be separated from highness in a similar fashion in the c.e. degrees?

  . . . is there an order-theoretic property which is shared by all a.e. dominating c.e. degrees but not all high c.e. degrees?

- The property of being cuppable was a good candidate, but failed (Barmpalias-Morphett 2007)

- A very attractive candidate is the property of forming minimal pair.
- There is a minimal pair of high c.e. degrees (Lachlan)
- Whether there is a minimal pair of c.e. a.e. dominating degrees is one of the very popular questions in the area

- The same can be asked for the Δ^0_2 substructure of the Turing degrees.
- **THEOREM** (B.): There is a minimal pair of a.e. dominating Δ^0_2 degrees.
- We do not have a plausible candidate for this case...

References

- Barmpalias, Tracing and domination in the Turing degrees, submitted 2008.
- Cole/Simpson, Mass problems and hyperarithmeticity, *J. Math. Log.*, 7(2):125–143, 2007.
- Nies, Computability and Randomness, Oxford Press 2009
- Downey and Hirschfeldt, Algorithmic randomness and complexity, Springer-Verlag, to appear.
- Webpage: `http://www.barmpalias.net`

Thank you!