Termination

Higher-Order Orderings

Predicates

- S[t] : t is terminating
- C[t] : t is computable

Computability

Inductive definition of C[t]:

- Basic t: C[t] if S[t]
- Arrow t: C[t] if C[t(s)] for all computable s (of the right type)

Lemmas

0. Reducts of computable terms are computable
1. Computable terms are terminating
2. Applications are computable if all reducts are

Main. Computable substitutions yield computable terms

Lemma 0

- Reducts of computable terms are computable

$$C[t] \ \& \ t \rightarrow u \Rightarrow C[u]$$

Proof of Lemma 0

$C[t] \ \& \ t \rightarrow u \Rightarrow C[u]$

- Induction on type
- Basic t: C[u] if S[u] if S[t] if C[t]
- Arrow t:$\sigma \rightarrow \tau$: By def, C[t(s):τ] for all computable s. By ind, C[u(s):τ], for all s. By def, C[u].

Lemma 1

- Computable terms are terminating

$$C[t] \Rightarrow S[t]$$

Proof of Lemma 1

C[t] ⇒ S[t]

- Induction on type
- Basic t: By definition
- Arrow t:σ→τ

  By def, C[t(s)] for all computable s:σ.

  By ind, S[t(s):τ]. It must be that S[t], too.

Neutrality

- applying creates no new redexes

  t neutral: redexes of t(s) are in t or s

- computable if reducts are

  C[t] if C[r] for all r s.t. t ➙ r

Lemma 2

Applications are neutral:

C[s(t)] if C[r] for all r s.t. s(t) ➔ r

Proof of Lemma 2

$C[s(t)]$ if $\forall r.\ s(t) \rightarrow r \Rightarrow C[r]$

- Induction on type of $s(t)$
- Basic: $S[s(t)]$ iff $S[r]\ \forall r$
- Arrow: Show $C[s(t)(u)]$ for each computable u.

  By ind, $C[r(u)]\ \forall r$ suffices, which is just $C[r]$.

Corollary

$C[(\lambda x.s)\ (t)]$ if $C[s\{x \mapsto t\}]$ & $C[t]$

By well-founded induction on s,t

Proof of Corollary

$C[s\{x \mapsto t\}]$ & $C[t] \Rightarrow C[(\lambda x.s)(t)]$

By L0, $S[s]$ & $S[t]$. Let $s \rightarrow s'$, $t \rightarrow t'$

So $C[s'\{x \mapsto t\}]$ & $C[t] \Rightarrow C[(\lambda x.s')(t)]$

$C[s\{x \mapsto t\}]$ & $C[t'] \Rightarrow C[(\lambda x.s)(t')]$

By L2, $C[(\lambda x.s)(t)]$ if $C[(\lambda x.s')(t)]$ & $C[(\lambda x.s)(t')]$ & $C[s\{x \mapsto t\}]$

But $C[t] \Rightarrow C[t']$ and $C[s\{x \mapsto t\}] \Rightarrow C[s'\{x \mapsto t\}]$

Lemma 3

S[t1] & … & S[tn] ⇒ C[x(t1)(t2)…(tn)]

- Induction on type of t = x(t1)(t2)…(tn)
- Basic t: Since only reducible inside terminating ti, S[t]. By def, C[t].
- Arrow t:$\sigma \rightarrow \tau$. For any computable s:σ, S[s] by L1. By ind, C[t(s):τ]. By def, C[t].

Main Lemma

- Computable substitutions yield computable terms

  Main: $C[u\sigma]$ for all u and computable σ

  - where $C[\sigma]$ if $C[t]$ for all $x \mapsto t$ in σ

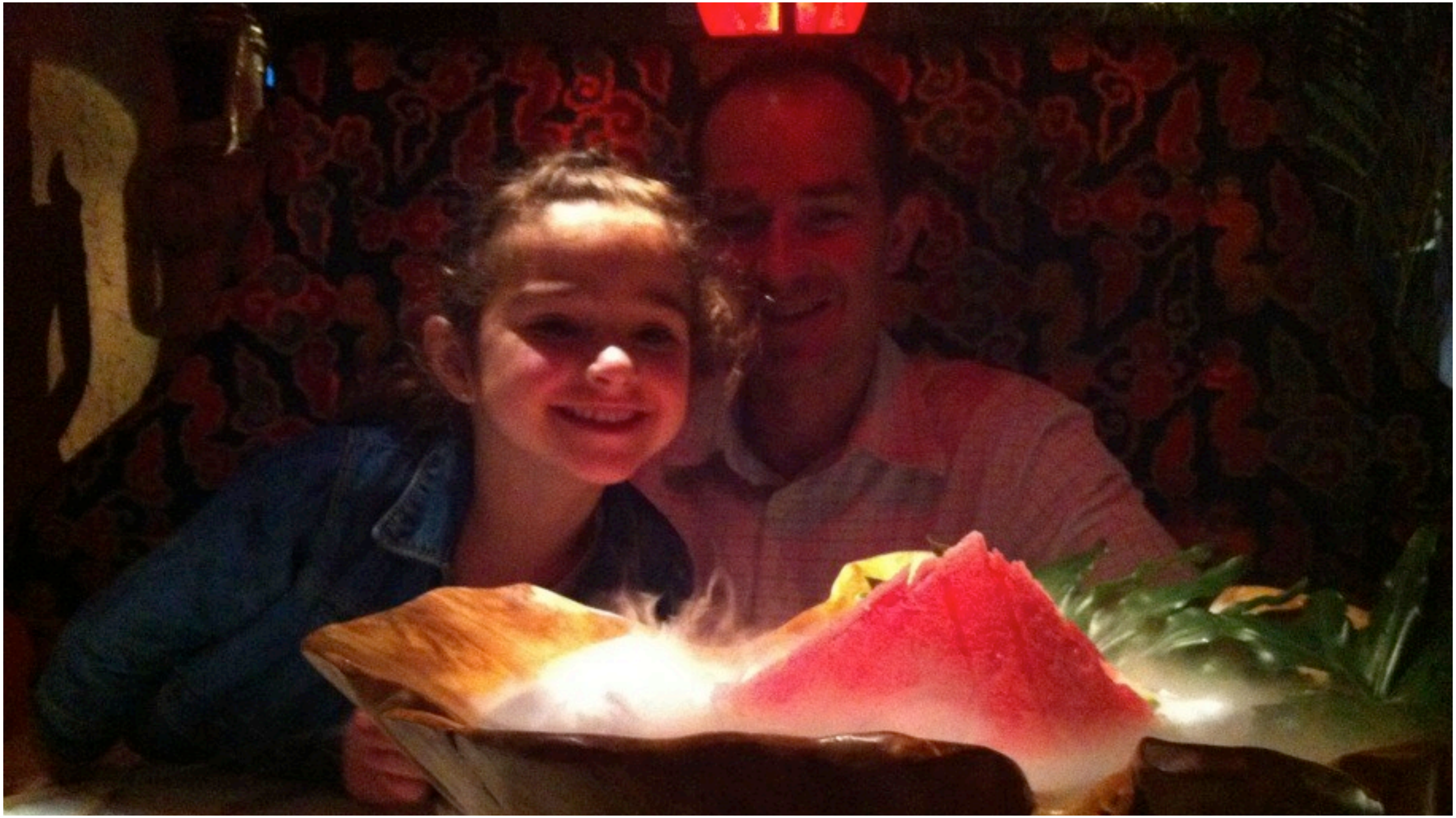
Proof of Main Lemma

$C[u\sigma]$ for computable σ

- Structural induction on u
- u constant: $u=u\sigma$ is basic and terminating; so $C[u]$ by def.
- u is variable x: If $x\sigma=x$, L3 applies; otherwise $x\sigma$ is computable.
- $u=t(s)$: $u\sigma=t\sigma(s\sigma)$. By ind, $C[t\sigma]$; by def, $C[t\sigma(s\sigma)]$, since $C[s\sigma]$ by ind.
- $u=\lambda x.s$: For computable t, let $\sigma'=\sigma-\{x\mapsto x\sigma\}\cup\{x\mapsto t\}$. By ind, $C[s\sigma']$. By L2c, $C[((\lambda x.s)\sigma)(t)]$, as $(\lambda x.s)\sigma = \lambda x.s(\sigma-\{x\mapsto x\sigma\})$ and $s(\sigma-\{x\mapsto x\sigma\})\{x\mapsto t\} = s\sigma'$. By def, $C[(\lambda x.s)\sigma]$.

Theorem

- All typed terms are terminating
    - C[t] for all t
        - Main lemma (empty substitution)
    - S[t] for all t
        - By Lemma 1



Frédéric

Functional

- $D(\lambda x.y) \to \lambda x.0$
- $D(\lambda x.x) \to \lambda x.1$
- $D(\lambda x.\sin(F(x))) \to \lambda x.D(F(x)) \cdot \cos(F(x))$

Higher-Order Rewriting

- map(F,e) ➔ e
- map(F,x:*y*) ➔ F(x):map(F,*y*)

System T

- $\mathrm{rec}(0,u,F) \rightarrow u$
- $\mathrm{rec}(s(x),u,F) \rightarrow F(x,\mathrm{rec}(x,u,F))$

- $n! \rightarrow \mathrm{rec}(n,1, \lambda y,z.s(y)\cdot z)$

Mixing Problem

- $f(c(x)) \rightarrow x$
  - $f: A \rightarrow (A \rightarrow B)$ $c: (A \rightarrow B) \rightarrow A$ $x: A \rightarrow B$
- $w = \lambda z{:}A.f(z)(z)$
  - $w(c(w)) \rightarrow f(c(w))(c(w)) \rightarrow w(c(w)) \rightarrow$

Explicit Application

- $@(s,t)$ for $s(t)$
- $@(F,t)$ for $F(t)$

System T

- rec(0,u,F) ➔ u
- rec(s(x),u,F) ➔ @(F,x,rec(x,u,F)))

Eta

- $\lambda x.f(x) =_\eta f$ (for $x \notin f$)
- eta long: $\lambda x.f(x)$

Higher-Order RPO

- precedence >
  - @ minimal
  - assume total (for simplicity)
- type order >
  - various conditions

Example Type Order

- $\sigma \to \tau > \tau$
- $\sigma \to \tau > a \Leftrightarrow \tau \geq a$ (base a)
- $\sigma \to \tau > \sigma' \to \tau' \Leftrightarrow \tau > \tau' \vee \sigma \geq \sigma' \to \tau'$
- well-founded even when enriched with $\sigma \to \tau > \sigma$

Higher-Order RPO

- $> \; = \; >^{\emptyset}$
- $>^{X}$ (keep track of variables X)
- $>^{X} \; = \; >^{X} \cap \geq$

Plain Cases

- $s = f(s1,\ldots,sm) >^X g(t1,\ldots,tn)$
  - if $f > g$ & $s >^X t1,\ldots,tn$
- $s = f(s1,\ldots,sm) >^X f(t1,\ldots,tn)$
  - if $\{s1,\ldots,sm\} > \{t1,\ldots,tn\}$ and $s >^X t1,\ldots,tn$
- $s = f(s1,\ldots,sm) >^X t$
  - if some $si \gtrsim^X t$

Variable Case

- $s >^{\{\ldots x \ldots\}} x$
  - if $s \neq x$

Lambda Cases

- $\lambda x{:}\alpha.w[x] >^X t$
  - if $w[z{:}\alpha] \gtrsim^X t$
- $s >^X \lambda y{:}\beta.w[y]$
  - if $s >^{X \cup \{z:\beta\}} w[z]$

Beta-Eta Cases

- $\lambda x.@(v,x) >^X t$
  - if $x \notin v$, $v \geq^X t$
- $@(\lambda x.w[x],v) >^X t$
  - if $w[v] \geq^X t$

Lambda-Lambda

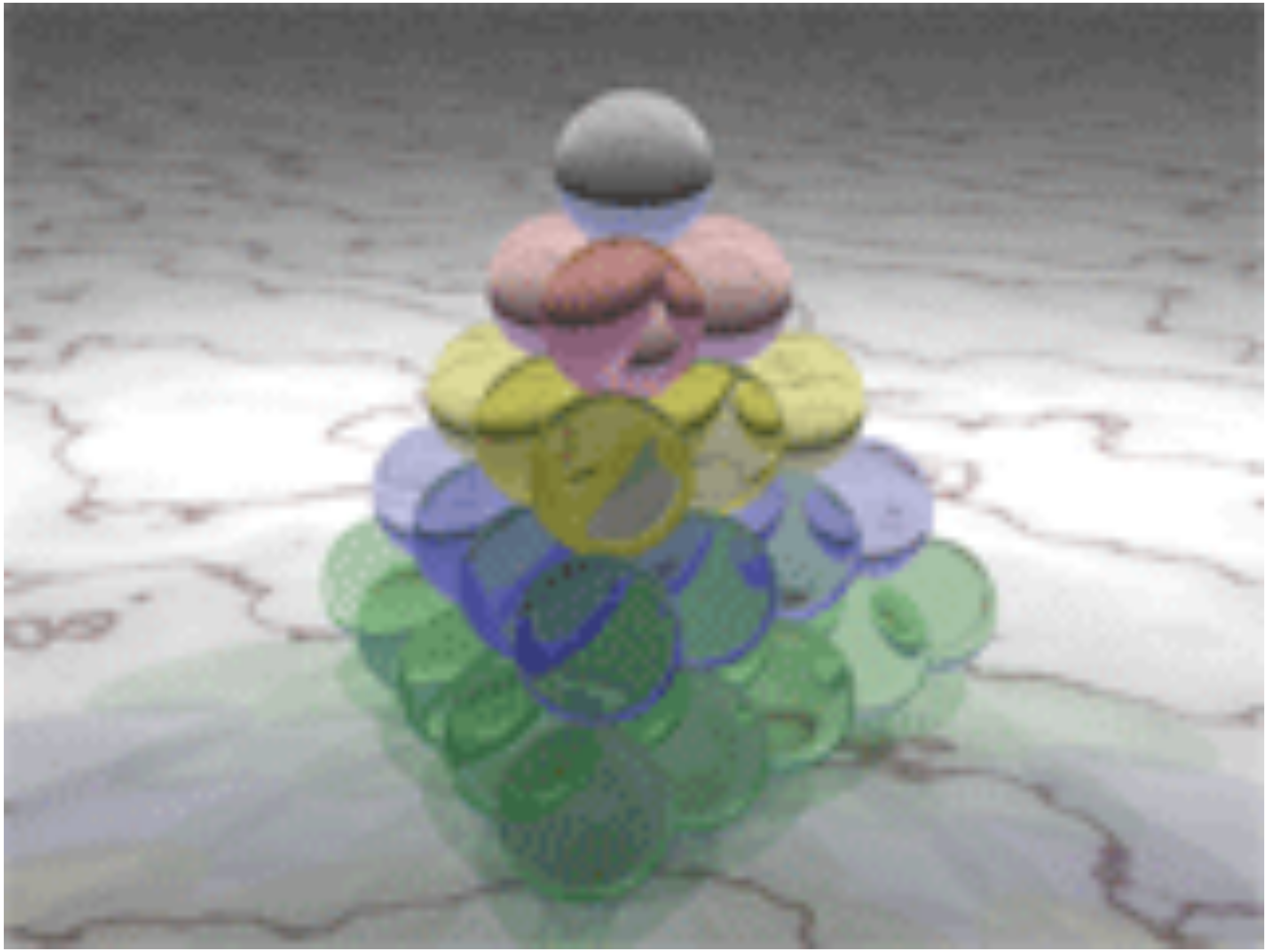
- $\lambda x{:}\alpha.u[x] >^X \lambda y{:}\alpha.w[y]$
  - if $u[z{:}\alpha] >^X w[z]$
- $s = \lambda x{:}\alpha.u[x] >^X \lambda y{:}\beta.w[y]$
  - if $\alpha \neq \beta$ & $s >^X w[z{:}\beta]$

System T

- rec(0,u,F) ➔ u
- rec(s(x),u,F) ➔ @(F,x,rec(x,u,F)))

Brower Ordinals

- $rec(0,U,V,W) \rightarrow U$
- $rec(s(X),U,V,W) \rightarrow @(V,X,rec(X,U,V,W))$
- $rec(lim(F), U, V, W) \rightarrow$
  $@(W, F, \lambda n.rec(@(F, n), U, V, W))$
- a little more needed



Kepler Conjecture

This is really the end