Core type theory

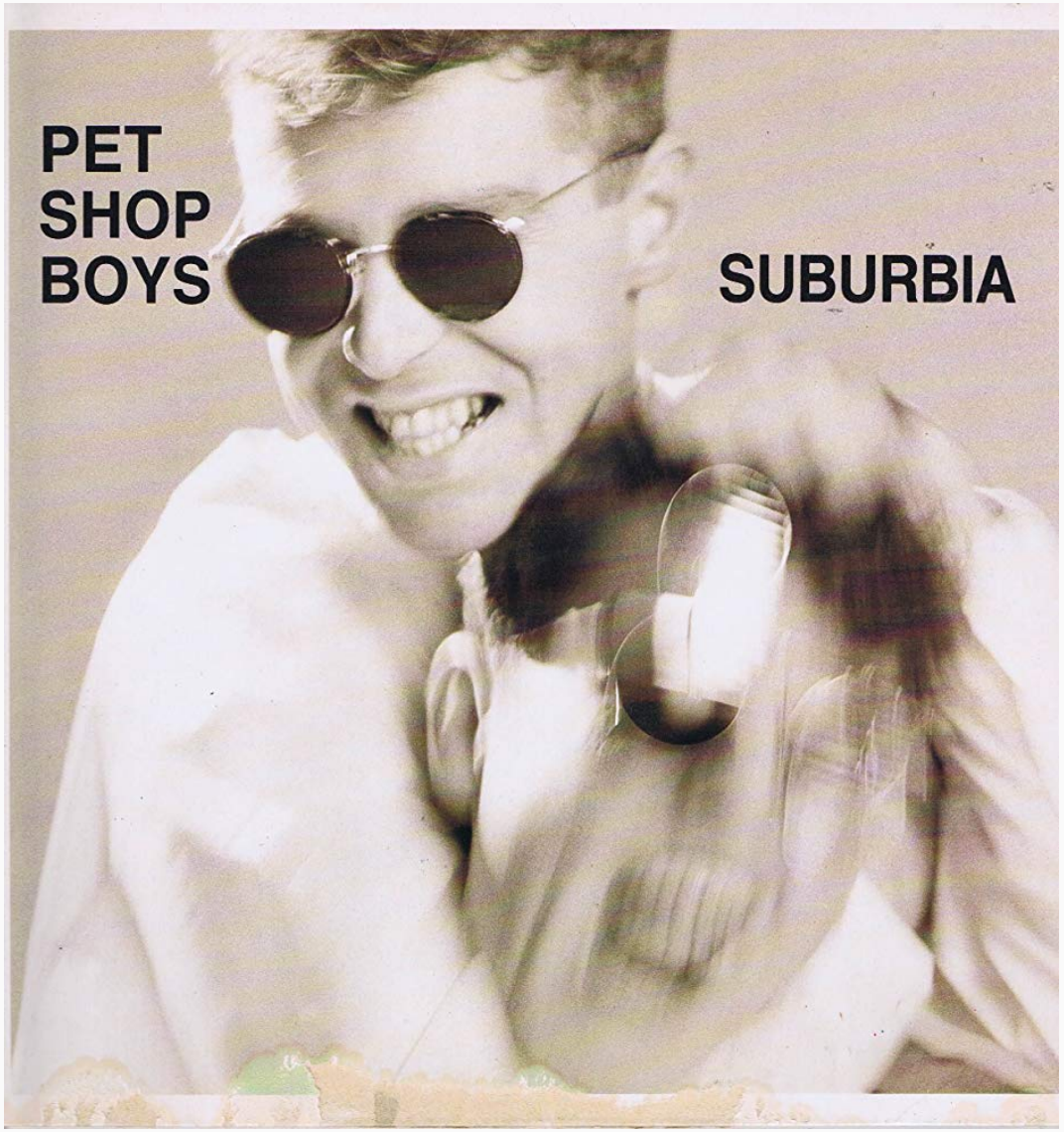
David Ripley

Monash University
http://davewripley.rocks

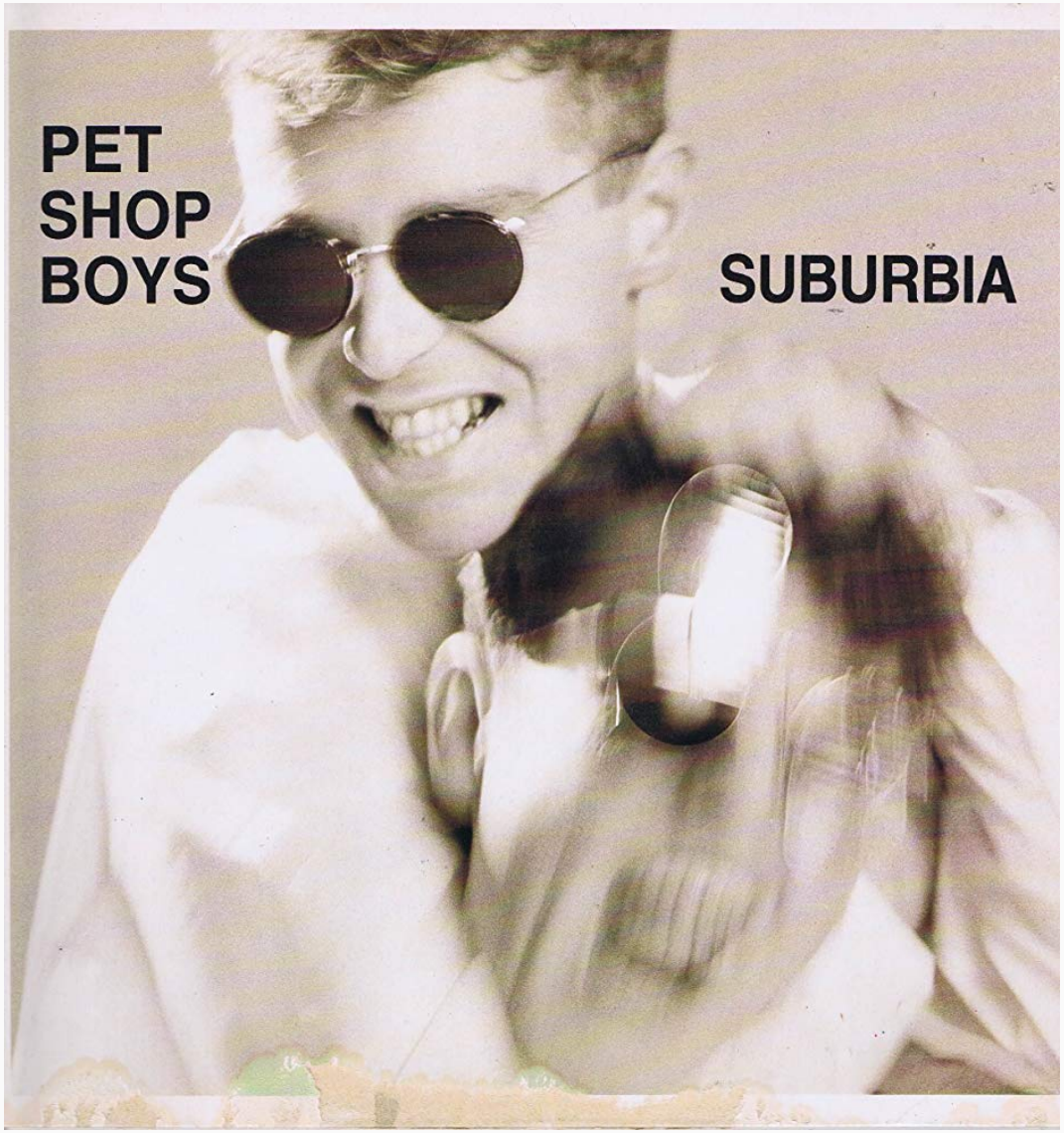
Core logic

Core logic (aka ‘intuitionistic relevant logic’)
is a system of logic devised and developed by Neil Tennant
over the last 40ish years.

No, not that Neil Tennant



No, not that Neil Tennant



(At least I don't think so.)

$$\rightarrow\mathrm{I}^n\colon \quad \dfrac{\begin{array}{c}[\varphi]^n\\ \vdots\\ \psi\end{array}}{\varphi \rightarrow \psi} \qquad \rightarrow\mathrm{I!}^n\colon \quad \dfrac{\begin{array}{c}[\varphi]^n\\ \vdots\\ \frownie\end{array}}{\varphi \rightarrow \psi} \qquad \rightarrow\mathrm{E}\colon \quad \dfrac{\varphi \rightarrow \psi \qquad \varphi \qquad \begin{array}{c}[\psi]^n\\ \vdots\\ \theta\end{array}}{\theta}$$

$$\neg\mathrm{I}^n\colon \quad \dfrac{\begin{array}{c}[\varphi]^n\\ \vdots\\ \frownie\end{array}}{\neg\varphi} \qquad \neg\mathrm{E}\colon \quad \dfrac{\neg\varphi \qquad \varphi}{\frownie}$$

In $\rightarrow$I and $\rightarrow$E, discharge may be vacuous.

In $\rightarrow$I! and $\neg$I, discharge must not be vacuous.

As it stands, that's a proof system for intuitionistic logic.

With the additional restriction that
major premises for elimination must be assumptions,
we get to core logic.

(Caveat: this isn't exactly how Tennant does it.
His proofs are more restrictive.
But all the same things are provable.)

Let an argument $\Gamma \succ \mathfrak{C}$ be core valid iff:
there is a core proof of $\mathfrak{C}$ whose open assumptions are all in Γ.

Then core validity is not closed under cut:
$\neg\varphi \succ \varphi \to \psi$ and $\varphi \to \psi, \varphi \succ \psi$ are both core valid,
but $\neg\varphi, \varphi \succ \psi$ is not.

Rather, $\neg\varphi, \varphi \succ ☹$ is.

Core validity is closely related to intuitionistic validity:

Results (Tennant):

- $\Gamma \vdash_{Core} \lightning$ iff $\Gamma \vdash_{\mathsf{Int}} \bot$
- If $\Gamma \nvdash \lightning$, then $\Gamma \vdash_{\mathsf{Int}} \varphi$ iff $\Gamma \vdash \varphi$

The difference is in what follows from inconsistent premise sets.

Curry-Howard

The simply-typed lambda calculus encodes a theory of typed data and functions on that data.

This is importantly connected to intuitionistic logic, via the Curry-Howard correspondence.

Our types are atoms, or $\bot$, or formed by $\to$ from other types.

$\bot$ is interpreted as an empty type,

and $\varphi \to \psi$ the type of functions
that take an input of type φ and give an output of type ψ.

There are variables $x^\varphi, y^\varphi, z^\varphi, \ldots$ of each type φ,

given terms $M^{\varphi\to\psi}$ and N^φ, there is $(MN^\varphi)^\psi$,

given a variable x^φ and a term M^ψ, there is $(\lambda x.M)^{\varphi\to\psi}$,

given a term $M^\bot$, there is $(\texttt{explode}\ M)^\varphi$.

There are variables $x^\varphi, y^\varphi, z^\varphi, \ldots$ of each type φ,
(variables are assumptions)

given terms $M^{\varphi\to\psi}$ and N^φ, there is $(MN^\varphi)^\psi$,

given a variable x^φ and a term M^ψ, there is $(\lambda x.M)^{\varphi\to\psi}$,

given a term $M^\bot$, there is $(\texttt{explode}\ M)^\varphi$.

There are variables $x^\varphi, y^\varphi, z^\varphi, \ldots$ of each type φ,
(variables are assumptions)

given terms $M^{\varphi\to\psi}$ and N^φ, there is $(MN^\varphi)^\psi$,
(function application is modus ponens)

given a variable x^φ and a term M^ψ, there is $(\lambda x.M)^{\varphi\to\psi}$,

given a term $M^\bot$, there is $(\texttt{explode}\ M)^\varphi$.

There are variables $x^{\varphi}, y^{\varphi}, z^{\varphi}, \ldots$ of each type φ,
(variables are assumptions)

given terms $M^{\varphi \to \psi}$ and N^{φ}, there is $(MN^{\varphi})^{\psi}$,
(function application is modus ponens)

given a variable x^{φ} and a term M^{ψ}, there is $(\lambda x.M)^{\varphi \to \psi}$,
(function formation is conditional proof;
variable binding is discharging an assumption)

given a term $M^{\bot}$, there is $(\texttt{explode}\ M)^{\varphi}$.

There are variables $x^{\varphi}, y^{\varphi}, z^{\varphi}, \ldots$ of each type φ,
(variables are assumptions)

given terms $M^{\varphi \to \psi}$ and N^{φ}, there is $(MN^{\varphi})^{\psi}$,
(function application is modus ponens)

given a variable x^{φ} and a term M^{ψ}, there is $(\lambda x.M)^{\varphi \to \psi}$,
(function formation is conditional proof;
variable binding is discharging an assumption)

given a term $M^{\bot}$, there is $(\texttt{explode}\ M)^{\varphi}$.
(crying uncle is explosion)

Any term of the form $((\lambda x.M)N^{\varphi})^{\psi}$ is a redex;
its reduct is $M[x \mapsto N]^{\psi}$.

Given a term with a selected redex as a subterm,
we can reduce the whole term by one step
by replacing the redex with its reduct,
leaving the context alone.

The result is always well-formed.

One-step reduction $\triangleright_{1\beta}$ and its reflexive transitive closure $\triangleright_\beta$ have some nice properties:

Type Preservation

If $M^\varphi \triangleright_\beta N$, then N^φ

Confluence

If $M \triangleright_{1\beta} N$ and $M \triangleright_{1\beta} O$, then there is a P with $N \triangleright_\beta P$ and $O \triangleright_\beta P$

Strong Normalization

All reduction paths are finite

Together, these mean that every M^{φ} has a unique normal form N^{φ}, which can be reached by reducing it willy-nilly until it can't be reduced farther.

This supports thinking of M^{φ} as a program for calculating a φ, normal forms as the values calculated, and reduction as program execution.

Core type theory

Again, propositions serve as types.

There is no $\bot$, and ☹ is not a proposition/type.

$\to$ is as before

$\neg\varphi$ is the type of things that take input of type φ and then crash.

There are variables $x^\varphi, y^\varphi, z^\varphi, \ldots$ of each type φ,

given terms $M^{\varphi\to\psi}$ and N^φ, there is $(MN^\varphi)^\psi$,

given a variable x^φ and a term M^ψ, there is $(\lambda x.M)^{\varphi\to\psi}$,

given a variable x^{φ} and a term $M^{\frownie}$ in which x occurs,
there is $(\lambda x.M)^{\neg\varphi}$ and $(\lambda x.M^{\frownie})^{\varphi\to\psi}$,

given $M^{\neg\varphi}$ and N^{φ}, there is $(MN^{\varphi})^{\frownie}$.

given a variable x^{φ} and a term $M^{\frownie}$ in which x occurs,
there is $(\lambda x.M)^{\neg\varphi}$ and $(\lambda x.M^{\frownie})^{\varphi\to\psi}$,
if it crashes, we can note that it crashes,
and if it crashes, we can pretend it outputs any ψ

given $M^{\neg\varphi}$ and N^{φ}, there is $(MN^{\varphi})^{\frownie}$.

given a variable x^{φ} and a term $M^{\frownie}$ in which x occurs,
there is $(\lambda x.M)^{\neg\varphi}$ and $(\lambda x.M^{\frownie})^{\varphi\to\psi}$,
if it crashes, we can note that it crashes,
and if it crashes, we can pretend it outputs any ψ

given $M^{\neg\varphi}$ and N^{φ}, there is $(MN^{\varphi})^{\frownie}$.
with the right input, we can make it crash

Redexes and their reducts are ‘as before’.

A redex is anything of the form $(\lambda x.M)N$,
and its reduct is $M[x \mapsto N]$.

	Redex	Reduct
•	$((\lambda x.M^{\psi})^{\varphi\to\psi}N^{\varphi})^{\psi}$	$M[x\mapsto N]^{\psi}$
•	$((\lambda x.M^{☹})^{\varphi\to\psi}N^{\varphi})^{\psi}$	$M[x\mapsto N]^{☹}$
•	$((\lambda x.M^{☹})^{\neg\varphi}N^{\varphi})^{☹}$	$M[x\mapsto N]^{☹}$

	Redex	Reduct
•	$((\lambda x.M^{\psi})^{\varphi\to\psi}N^{\varphi})^{\psi}$	$M[x \mapsto N]^{\psi}$
•	$((\lambda x.M^{☹})^{\varphi\to\psi}N^{\varphi})^{\psi}$	$M[x \mapsto N]^{☹}$
•	$((\lambda x.M^{☹})^{\neg\varphi}N^{\varphi})^{☹}$	$M[x \mapsto N]^{☹}$

Reduction does not always preserve type!

	Redex	Reduct
•	$((\lambda x.M^{\psi})^{\varphi\to\psi}N^{\varphi})^{\psi}$	$M[x \mapsto N]^{\psi}$
•	$((\lambda x.M^{☹})^{\varphi\to\psi}N^{\varphi})^{\psi}$	$M[x \mapsto N]^{☹}$
•	$((\lambda x.M^{☹})^{\neg\varphi}N^{\varphi})^{☹}$	$M[x \mapsto N]^{☹}$

Reduction can remove free variables!

So one-step reduction of a term at a redex is not so simple.

Just replacing the redex with its reduct, leaving the context alone, is not always well-formed. (!)

The solution: leave the context alone when you can, and otherwise discard what you must.

- If $M^{\varphi} \rhd_{1\beta} M'^{\varphi}$, then $MN \rhd_{1\beta} M'N$
- If $M \rhd_{1\beta} M'^{\frownie}$, then $MN \rhd_{1\beta} M'$

- If $N^{\varphi} \rhd_{1\beta} N'^{\varphi}$, then $MN \rhd_{1\beta} MN'$
- If $N \rhd_{1\beta} N'^{\frownie}$, then $MN \rhd_{1\beta} N'^{\frownie}$

- If $M^{\varphi} \rhd_{1\beta} M'^{\varphi}$, then $\lambda x.M \rhd_{1\beta} \lambda x.M'$
- If $M \rhd_{1\beta} M'^{\frownie}$,
  - if $x \in \mathbf{FV}(M')$, then $\lambda x.M \rhd_{1\beta} \lambda x.M'$ (preserving hat)
  - if $x \notin \mathbf{FV}(M')$, then $\lambda x.M \rhd_{1\beta} M'$

Example

$\left((\lambda y^{\varphi}.(x^{\neg\varphi} y^{\varphi})^{\frowny})^{\varphi\to\theta} z^{\varphi}\right)^{\theta}$ is a redex, and it reduces in one step to $(x^{\neg\varphi} z^{\varphi})^{\frowny}$.

Example

Let M be the above redex, and let M' be its reduct. Then $(\lambda w^{\rho}.M^{\theta})^{\rho\to\theta} \rhd_{1\beta} M'^{\frowny}$.

Example

With the same M and M', we have $\left(\lambda z^{\varphi}.(\lambda w^{\rho}.M^{\theta})^{\rho\to\theta}\right)^{\varphi\to\rho\to\theta} \rhd_{1\beta} (\lambda z^{\varphi}.M'^{\frowny})^{\varphi\to\rho\to\theta}$.

Results

Reduction does not preserve type.

But: it can only change from a type to ☹

Never from one type to another, or from ☹ to a type

Reduction is not confluent.

Example

$$\left(\lambda v^{\rho}.\left((\lambda x^{\psi}.y^{\theta})^{\psi\to\theta}\quad\left((\lambda u^{\neg\rho}.(u^{\neg\rho}v^{\rho})^{\frownie})^{\neg\rho\to\psi}z^{\neg\rho}\right)^{\psi}\right)^{\theta}\right)^{\rho\to\theta}$$

reduces in one step to $(\lambda v^{\rho}.y^{\theta})^{\rho\to\theta}$, or to $(\lambda v^{\rho}.(z^{\neg\rho}v^{\rho})^{\frownie})^{\rho\to\theta}$

Example

$$\left(\left((\lambda v^{\theta}.(u^{\neg\theta}v^{\theta})^{\Sadness})^{\theta\to\varphi\to\psi}t^{\theta}\right)^{\varphi\to\psi}\left((\lambda x^{\rho}.(y^{\neg\rho}x^{\rho})^{\Sadness})^{\rho\to\varphi}w^{\rho}\right)^{\varphi}\right)^{\psi}$$

reduces in one step to $(u^{\neg\theta}t^{\theta})^{\Sadness}$, or to $(y^{\neg\rho}w^{\rho})^{\Sadness}$

Let $=_\beta$ be the least equivalence relation including $\triangleright_\beta$.

Then for all terms M, N, regardless of type, $M =_\beta N$. (!)

It might be nice to find a weakening of confluence
that reduction does obey.

It's definitely not:
confluent on closed terms,
confluent-but-for-☹s,
confluent on atomic types.

But is it confluent-but-for-☹s on closed terms of atomic type?
Is there a confluent reduction strategy?

Reduction is still strongly normalizing.

Every reduction path is finite.

The proof is an old one
(newfangled proofs don't work here!):

define a notion of 'strongly computable term' by induction on types,
then show simultaneously that all SC terms are SN
and that all terms are SC.

So core ‘programs’ always terminate.

But whether they crash,
and what result they produce if they don’t,
can depend on choices made in execution.

Wrapup

Core logic is not very far from intuitionistic logic.

But their associated type theories are very different.

Preservation and confluence do not hold in core type theory, although strong normalization still does.

Girard has said that a logic without cut elimination
is ‘like a car without [an] engine’.

Core type theory seems to show that this is wrong:
despite the lack of cut elimination, computation proceeds as usual.

(This is basically what Tennant’s been saying all along.)

Core logic, it turns out, is more like a car without a steering wheel.