Notes from August 25, 2017

September 7, 2017

Hirsch and Tate [2017] describe a way of giving semantics to languages with both a consumer effect (or coeffect) and a producer effect. Brunel et al. [2014] describe using multiple consumer effects in the form of a graded comonad to do usage analysis. Finally, Gaboardi et al. [2016] describe a way of combining a graded comonad and a graded monad in the same language using distributive laws. However, Hirsch and Tate [2017] describe how distributive laws do not give semantics to some languages with only one consumer effect and one producer effect. This project attempts to use the technique of Hirsch and Tate [2017] with the graded monads and comonads of Brunel et al. [2014] and Gaboardi et al. [2016].

Importantly, in Gaboardi et al. [2016]'s work, the producer effects and the consumer effects interact via two functions which describe how both consumer and producer effects change via the distributive law. At first glance, it is less clear that they interact in a similar way in Hirsch and Tate [2017]'s work, where the effects are described as interacting via *timing*: either the producer effect is resolved first, or the consumer effect is. However, if we use an example of classical linear logic, we get the type system in Figure 1, which has a Gaboardi et al. [2016]-style interaction.

The grading is done via two rings, $\langle \mathcal{R}, +, \cdot, 0, 1 \rangle$ which describe the consumer effects (and the grading on the comonad) and $\langle \mathcal{P}, +, \cdot, 0, 1 \rangle$ which describes the producer effects (and the grading on the monad). There are two functions $\mathcal{U} : \mathcal{R} \times \mathcal{P} \to \mathcal{P}$ and $\mathcal{V} : \mathcal{R} \times \mathcal{P} \to \mathcal{R}$. These lift naturally to vectors. (Note the use of colors to clarify types: all consumer effects/coeffect grades will be written in blue, while all producer effects/effect grades will be written in red.)

The functions $\mathcal{U}$ and $\mathcal{V}$ must satisfy certain laws for cut elimination to hold. We look at the cut elimination steps and the equations these force on $\mathcal{U}$ and $\mathcal{V}$ now.

$$\frac{}{\tau \vdash \tau}$$

$$\frac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \otimes \psi \vdash \Delta} \qquad \frac{\Gamma \vdash \varphi, \Delta \qquad \Gamma' \vdash \psi, \Delta}{\Gamma, \Gamma' \vdash \varphi \otimes \psi, \Delta, \Delta'}$$

$$\frac{\Gamma, \varphi \vdash \Delta \qquad \Gamma', \psi \vdash \Delta'}{\Gamma, \Gamma', \varphi \parr \psi \vdash \Delta, \Delta'} \qquad \frac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \parr \psi, \Delta}$$

$$\frac{\Gamma, \varphi \vdash \Delta}{\Gamma, !_1 \varphi \vdash \Delta} \qquad \frac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{r' \cdot \vec{r}}\Gamma \vdash !_{r'}\varphi, ?_{\mathcal{U}(r', \vec{e})}\Delta}$$

$$\frac{\Gamma, !_r \varphi, !_s \varphi \vdash \Delta}{\Gamma !_{r+s} \varphi \vdash \Delta} \qquad \frac{\Gamma \vdash \Delta}{\Gamma, !_0 \varphi \vdash \Delta}$$

$$\frac{\Gamma \vdash \varphi, \Delta}{\Gamma \vdash ?_1 \varphi, \Delta} \qquad \frac{!_{\vec{r}}\Gamma, \varphi \vdash ?_{\vec{e}}\Delta}{!_{\mathcal{V}(\vec{r}, e)}\Gamma, ?_e \varphi \vdash ?_{\vec{e} \cdot e}\Delta}$$

$$\frac{\Gamma \vdash ?_e \varphi, ?_{e'} \varphi, \Delta}{\Gamma \vdash ?_{e+e'} \varphi, \Delta} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash ?_0 \varphi, \Delta}$$

Figure 1: Graded Classical Linear Logic

1 Cutting Left Promotion

With Left Contraction Consider the case where we cut a left promotion with a left contraction:

$$\frac{\dfrac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r} \cdot (s_1 + s_2)}\Gamma \vdash !_{s_1 + s_2}\varphi, ?_{\mathcal{U}(s_1 + s_2, \vec{e})}\Delta} \qquad \dfrac{\Gamma', !_{s_1}\varphi, !_{s_2}\varphi \vdash \Delta'}{\Gamma', !_{s_1 + s_2}\varphi \vdash \Delta}}{!_{\vec{r} \cdot (s_1 + s_2)}\Gamma, \Gamma' \vdash ?_{\mathcal{U}(s_1 + s_2, \vec{e})}\Delta, \Delta'}$$

One application of cut-elimination gives:

$$\frac{\dfrac{\dfrac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r} \cdot s_2}\Gamma \vdash !_{s_2}\varphi, ?_{\mathcal{U}(s_2, e)}\Delta} \qquad \dfrac{\dfrac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r} \cdot s_1}\Gamma \vdash !_{s_1}\varphi, ?_{\mathcal{U}(s_1, e)}\Delta} \qquad \Gamma', !_{s_1}\varphi, !_{s_2}\varphi \vdash \Delta'}{!_{\vec{r} \cdot s_1}\Gamma, \Gamma', !_{s_2}\varphi \vdash ?_{\mathcal{U}(s_1, e)}\Delta, \Delta'}}{!_{\vec{r} \cdot s_1}\Gamma, !_{\vec{r} \cdot s_1}\Gamma, \Gamma' \vdash ?_{\mathcal{U}(s_1, e)}\Delta, ?_{\mathcal{U}(s_2, e)}\Delta, \Delta'}}{!_{\vec{r} \cdot s_1 + \vec{r} \cdot s_1}\Gamma, \Gamma' \vdash ?_{\mathcal{U}(s_1, e) + \mathcal{U}(s_2, e)}\Delta, \Delta'}$$

This gives rise to the following function equation:

$$\mathcal{U}(r_1 + r_2, \vec{e}) = \mathcal{U}(r_1, \vec{e}) + \mathcal{U}(r_2, \vec{e})$$

It also gives evidence that we need the following ring axiom:

$$r \cdot (s_1 + s_2) = (r \cdot s_1) + (r \cdot s_2)$$

With Left Weakening Cutting left promotion with left weakening gives:

$$\dfrac{\dfrac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r}\cdot 0}\Gamma \vdash !_0\varphi, ?_{\mathcal{U}(0,\vec{e})}\Delta} \qquad \dfrac{\Gamma' \vdash \Delta'}{\Gamma', !_0\varphi \vdash \Delta'}}{!_{\vec{r}\cdot 0}\Gamma, \Gamma' \vdash ?_{\mathcal{U}(0,\vec{e})}\Delta}$$

After one application of cut elimination, we get:

$$\dfrac{\Gamma' \vdash \Delta'}{!_0\Gamma, \Gamma' \vdash ?_0\Delta, \Delta'}$$

This gives rise to the equation

$$\mathcal{U}(0, \vec{e}) = 0$$

and also gives evidence that we need the ring identity

$$r \cdot 0 = 0$$

With Left Dereliction Consider cutting a left promotion with dereliction:

$$\dfrac{\dfrac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r}\cdot 1} \vdash !_1\varphi, ?_{\mathcal{U}(1,\vec{e})}\Delta} \qquad \dfrac{\Gamma', \varphi \vdash \Delta'}{\Gamma', !_1\varphi \vdash \Delta'}}{!_{\vec{r}\cdot 1}\Gamma, \Gamma' \vdash ?_{\mathcal{U}(1,\vec{e})}\Delta, \Delta'}$$

One application of the cut-elimination procedure gives:

$$\dfrac{!_{\vec{r}}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta \qquad \Gamma', \varphi \vdash \Delta}{!_{\vec{r}}\Gamma, \Gamma' \vdash ?_{\vec{e}}\Delta, \Delta'}$$

This shows that we need the equation

$$\mathcal{U}(1, \vec{e}) = \vec{e}$$

and that we need the ring identity

$$r \cdot 1 = 1$$

With Left Promotion Consider the case where we cut a left promotion with a left promotion:

$$\dfrac{\dfrac{!_{\vec{r}_1}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r}_1\cdot s_1\cdot s_2}\Gamma \vdash !_{s_1\cdot s_2}\varphi, ?_{\mathcal{U}(s_1\cdot s_2,\vec{e})}\Delta} \qquad \dfrac{!_{\vec{r}_2}\Gamma', !_{s_1}\varphi \vdash \psi, ?_{\vec{e}'}\Delta'}{!_{\vec{r}_2\cdot s_2}\Gamma', !_{s_1\cdot s_2}\varphi \vdash !_{s_2}\psi, ?_{\mathcal{U}(s_2,\vec{e}')}\Delta'}}{!_{\vec{r}_1}\Gamma, !_{\vec{r}_2}\Gamma' \vdash !_{s_2}\psi, ?_{\mathcal{U}(s_1\cdot s_2,\vec{e})}\Delta, ?_{\mathcal{U}(s_2,\vec{e}')}\Delta'}$$

One application of the cut-elimination procedure gives:

$$\dfrac{\dfrac{\dfrac{!_{\vec{r}_1}\Gamma \vdash \varphi, ?_{\vec{e}}\Delta}{!_{\vec{r}_1\cdot s_1}\Gamma \vdash !_{s_1}\varphi, ?_{\mathcal{U}(s_1,\vec{e})}\Delta} \qquad !_{\vec{r}_2}\Gamma', !_{s_1}\varphi \vdash \psi, ?_{\vec{e}'}\Delta'}{!_{\vec{r}_1\cdot s_1}\Gamma, !_{\vec{r}_2}\Gamma' \vdash \psi, ?_{\mathcal{U}(s_1,\vec{e})}\Delta, ?_{\vec{e}'}\Delta'}}{!_{\vec{r}_1\cdot s_1\cdot s_2}\Gamma, !_{\vec{r}_2\cdot s_2}\Gamma' \vdash !_{s_2}\psi, ?_{\mathcal{U}(s_2,\mathcal{U}(s_1,\vec{e}))}\Delta, ?_{\mathcal{U}(s_2,\vec{e}')}\Delta'}$$

This gives rise to the following equation:

$$\mathcal{U}(r_1 \cdot r_2, \vec{e}) = \mathcal{U}(r_2, \mathcal{U}(r_1, \vec{e}))$$

With Right Promotion Consider the case where we cut a left promotion with a right promotion:

$$\dfrac{\dfrac{!_r\Gamma \vdash \varphi, ?_e\psi, ?_{\vec{e}}\Delta}{!_{r\cdot r'}\Gamma \vdash !_{r'}\varphi, ?_{\mathcal{U}(r',e)}\psi, ?_{\mathcal{U}(r',\vec{e})}\Delta} \qquad \dfrac{!_s\Gamma', \psi \vdash ?_{\vec{e}'}\Delta'}{!_{\mathcal{V}(s,\mathcal{U}(r',e))}\Gamma', ?_{\mathcal{U}(r',e)}\psi \vdash ?_{\mathcal{U}(r',e)\cdot\vec{e}'}\Delta'}}{!_{r\cdot r'}\Gamma, !_{\mathcal{V}(s,\mathcal{U}(r',e))}\Gamma' \vdash !_{r'}\varphi, ?_{\mathcal{U}(r',\vec{e})}\Delta, ?_{\mathcal{U}(r',e)\cdot\vec{e}'}\Delta'}$$

One application of the cut-elimination procedure gives:

$$\dfrac{\dfrac{!_r\Gamma \vdash \varphi, ?_e\psi, ?_{\vec{e}}\Delta \qquad \dfrac{!_s\Gamma', \psi \vdash ?_{\vec{e}'}\Delta'}{!_{\mathcal{V}(s,e)}\Gamma', ?_e\psi \vdash ?_{e\cdot\vec{e}'}\Delta}}{!_r\Gamma, !_{\mathcal{V}(s,e)}\Gamma' \vdash \varphi, ?_{\vec{e}}\Delta, ?_{e\cdot\vec{e}'}\Delta}}{!_{r\cdot r'}\Gamma, !_{\mathcal{V}(s,e)\cdot r'}\Gamma' \vdash !_{r'}\varphi, ?_{\mathcal{U}(r',\vec{e})}\Delta, ?_{\mathcal{U}(r',e\cdot\vec{e}')}\Delta}$$

This gives rise to two equations:

$$\mathcal{V}(r, \mathcal{U}(r', e)) = \mathcal{V}(r, e) \cdot r'$$
$$\mathcal{U}(r, \vec{e}) \cdot e = \mathcal{U}(r, e \cdot \vec{e})$$

Summary To sum up, after the following reasoning (and dualizing by cutting with right promotion) we need the following:

- $\mathcal{U}(0, e) = 0$
- $\mathcal{U}(1, e) = e$
- $\mathcal{U}(r_1 + r_2, e) = \mathcal{U}(r_1, e) + \mathcal{U}(r_2, e)$
- $\mathcal{U}(r_1 \cdot r_2, e) = \mathcal{U}(r_2, \mathcal{U}(r_1, e))$
- $\mathcal{U}(r, e) \cdot e' = \mathcal{U}(r, e \cdot e')$
- $\mathcal{U}(\mathcal{V}(r, e'), e) = e' \cdot \mathcal{U}(r, e)$
- $\mathcal{V}(r, 0) = 0$
- $\mathcal{V}(r, 1) = r$
- $\mathcal{V}(r, e_1 + e_2) = \mathcal{V}(r, e_1) + \mathcal{V}(r, e_2)$
- $\mathcal{V}(r, e_1 \cdot e_2) = \mathcal{V}(\mathcal{U}(r, e_1), e_2)$
- $\mathcal{V}(r, e) \cdot r' = \mathcal{U}(r \cdot r', e)$
- $\mathcal{V}(r, \mathcal{U}(r', e)) = \mathcal{V}(r, e) \cdot r'$

2 Motivating Language Example

We want to think about a parallel language which uses network bandwidth both down and up. We use bandwidth down when we use an input, since we need to download the inputs from the network. We use bandwidth up when we produce and output, since we need to upload the outputs to the network.

In this case, we can use the linear logic type system described above (perhaps with extensions). Then, both $\mathcal{R}$ and $\mathcal{P}$ are the natural numbers with the standard interpretation of the natural numbers as a ring. Moreover $\mathcal{U}(r, e) = r \cdot e$ and $\mathcal{V}(r, e) = r \cdot e$. With this interpretation, the above equations are satisfied.

Note 9-7-2017 I wrote down something Marco mentioned earlier: that he has two types of cost-counting analyses. One uses a traditional Kleisli-style arrow

$$A \to M_c B$$

which calculates cost correctly in a call-by-value language. The other uses arrows of the form

$$M_{c_1} A \to M_{c_2} B,$$

which calculates cost in a call-by-name language.

It occurs to me that, since his call-by-name language almost certainly allows arbitrary weakening and contraction, that it exists in the coKleisli category for a ! operator. That is, the call-by-name arrows are actually of the form

$$!M_{c_1} A \to M_{c_2} B,$$

which is predicted by Hirsch and Tate [2017]. Since there does not seem to be a distributive law between ! and $?_n$, it is likely that the strict case acts over the functor category for !, so that the call-by-value arrows are actually of the form $!A \to M_c !B$, as predicted by Hirsch and Tate [2017]. However, this seems to indicate that the strict and lazy notions from Hirsch and Tate [2017] should generalize in this work.

3 Motivating Reasoning Example

Imagine an analysis where terms may be either terminating or possibly-non-terminating (as producer effects) and may be used any natural number of times (as consumer effects). Then, we would like to show that the term 3 terminates in a lazy context. In Hirsch and Tate [2017], we can prove this as a metatheorem. However, we would like to be able to read this off the type of a thunked program (in Hirsch and Tate [2017]'s parlance). This would be a proof of the typing judgment $x : !_0 ?_{\text{pnt}} \vdash 3 : ?_{\text{t}} \mathbb{N}$ where pnt stands for "possibly non-terminating" and "t" stands for "terminating."

Because $?_{\text{t}}$ is the identity functor, this is the same as a judgment of the form $x : !_0 ?_{\text{pnt}} \vdash 3 : \mathbb{N}$. This is provable in Hirsch and Tate [2017]'s type system, by applying weakening. However, the "analyzed" (thunked) programs do not reveal this, since every function is listed as possibly not terminating.

4 Roadmap

1. Work out the full example of graded classical linear logic, with a term calculus
2. Work out a more-general example where we do not assume a quantitative nature
3. Describe more examples where the sort of reasoning given above can make analyses more precise
4. What semantic properties of (graded) monads/producers and (graded) comonads/consumptors are we assuming to get the bind/promotion rules above?

References

Aloïs Brunel, Marco Gaboardi, Damiano Mezza, and Steve Zdancewic. A core quantitative coeffect calculus. In *ESOP*, 2014.

Marco Gaboardi, Shin-ya Katsumata, Dominic Orchard, Flavien Breuvart, and Tarmo Uustalu. Combining effects and coeffects via grading. In *ICFP*, 2016.

Andrew K. Hirsch and Ross Tate. Strict and lazy semantics for effects: Layering monads and comonads. Technical report, Cornell University Department of Computer Science, 2017.