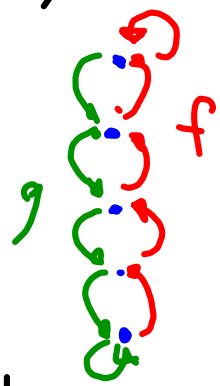
Escalator algebras

$$M_\mu = \langle \{0, 1, \ldots \mu\}, f_\mu, g_\mu \rangle$$

$$f_\mu(x) = \min(x+1, \mu)$$

$$g_\mu(x) = \max(x-1, 0)$$



The quasi-variety of M_μ is the smallest collection containing M_μ closed under powers, sub-algebras and isomorphism $ISP(M_\mu)$

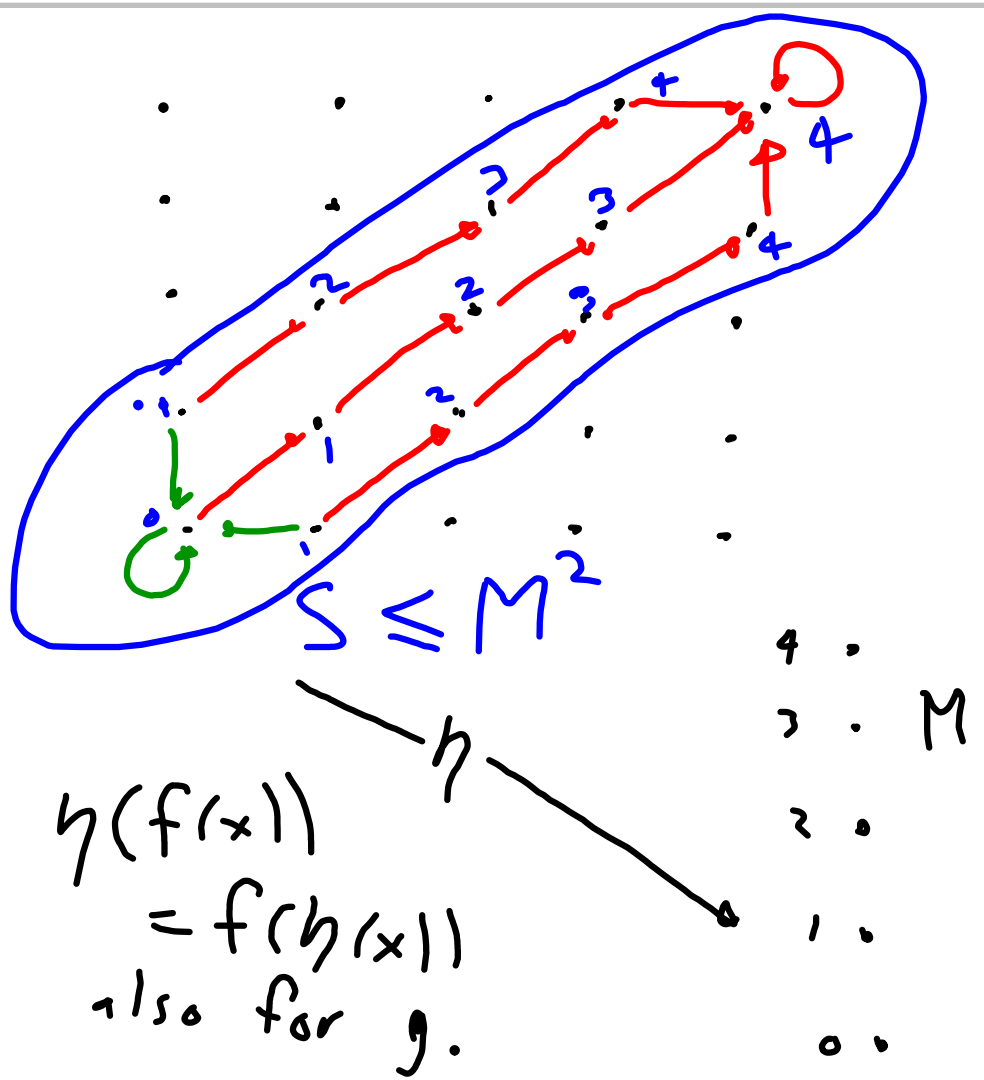
Thm

Let $S \leq M^n$ be a finite subalgebra of a power.

$\eta : S \to M$ be a hom.

Then η is locally a projection, meaning

$\forall a \in S \quad \eta(a) = a_i$ for some i.



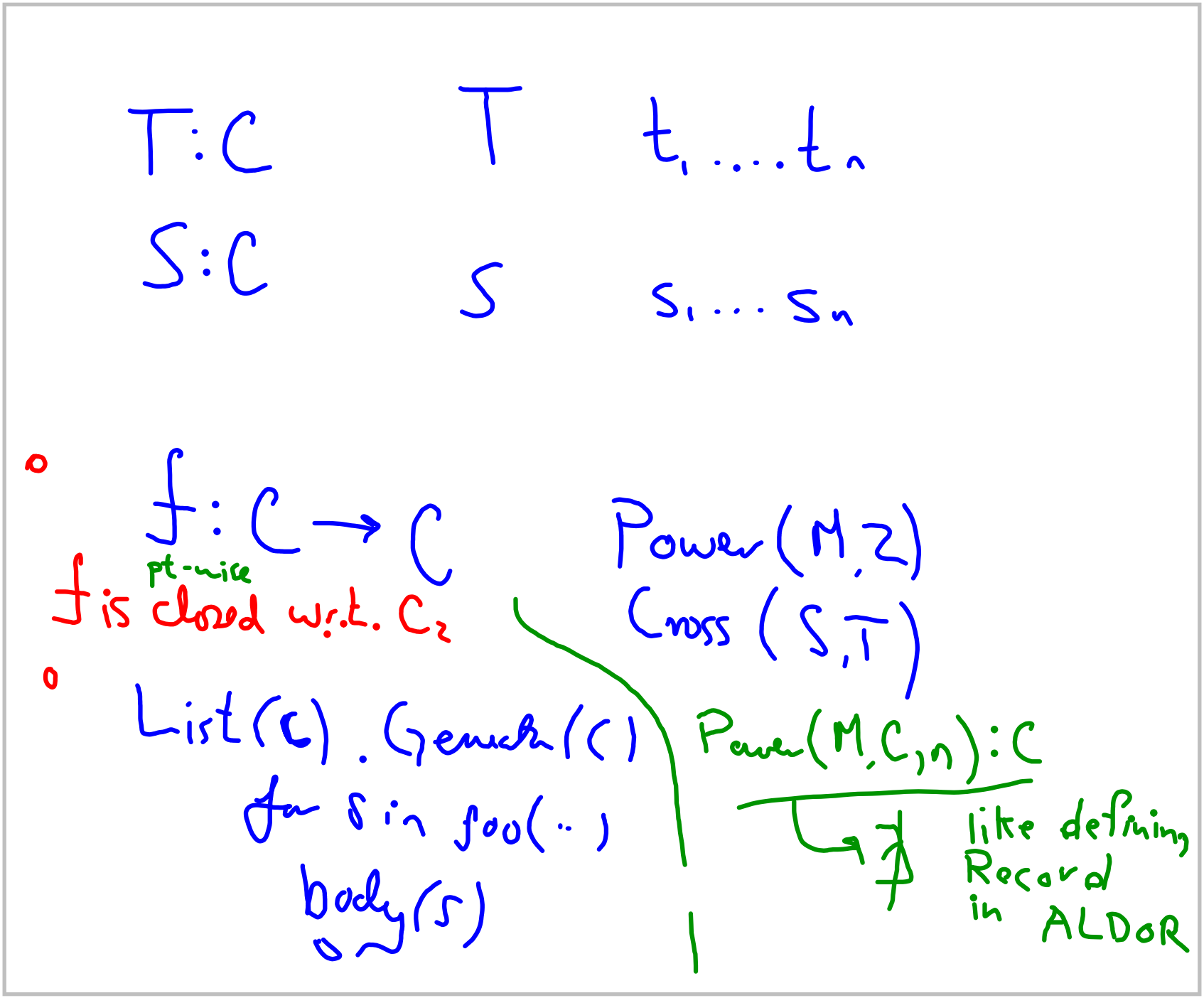
$\eta(f(x)) = f(\eta(x))$ also for g.

M : ? = Something(
SM4, [f, g]);

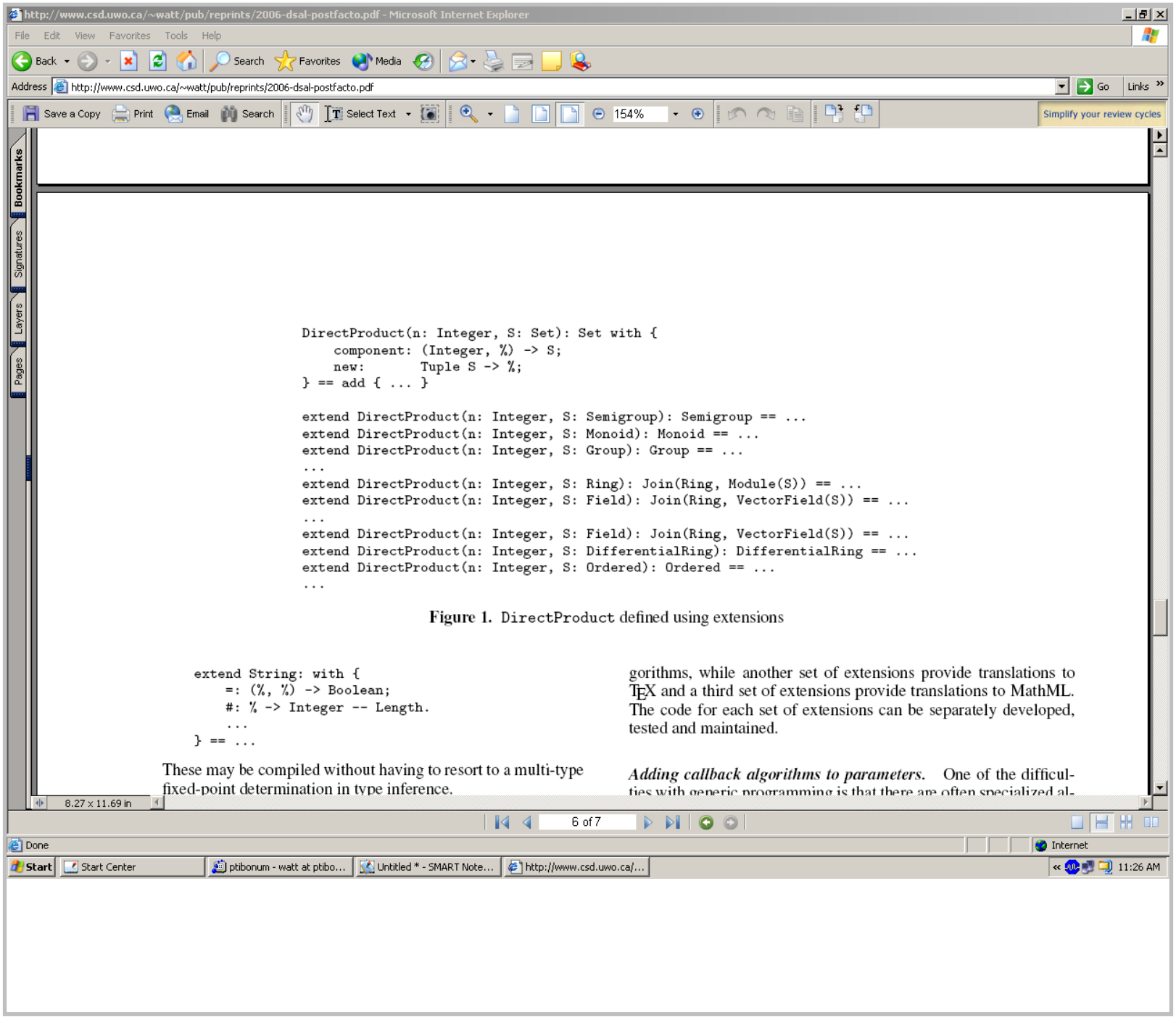
SM4 : Set Integer
== [0, 1, 2, 3, 4]

M2 == Power(M, 2)

for s in subalgebras(M2) -
repeat
|


T:C
T
t1 ... tn
S:C
S
s1 ... sn
f: C → C
pt-wise
f is closed w.r.t. C2
List(C) . Generator(C)
for s in foo(..)
body(s)
Power(M,Z)
Cross(S,T)
Power(M,C,n):C
like defining Record in ALDOR

```
body(F:FThingy): Z := {
    l : List(F) := elements()$F
    for e in l repeat {
        (u,v) := thy(e) -- F -> (F,F)
    return cont(...)
}
```



Title: Aug 23 - 11:26 AM (6 of 10)

- Conforming Type and the relation in Cayenne.

$S : Set(S, \ldots\ldots)$

- Early Binding of →, =, Integer, Boolean
which ones? Is this the language?
- Related to binding of \ / as extend. QG
min lim
- Is this decl
(M) : Module(M. S. + - * . .)
or is this
Them 1 : Module(M, R, ...) | F(M ... ∃ Thm : Module(M. R.)
- # of Params

R : Ring

P : UnivPolyCat(R)

E : SimpleAlgebraicExtension(p : P,
P : UnivPolyCat(R),
R : Ring)

#params = ht of tree

Size of expr is $O(ht^2)$

|Mod(R, M)|

— Perhaps
Module M
∃! R st Module (M, R, +, ·)

is like
Group G
∃! \ st QG (G, *, \, /, ..)

---

① Then → Integer Boolean
don't need to be spaced
either bound
or default

② Marc doesn't get RSI (Type Exp O(n))

③ op renaming, becomes tactical

implications?

=> – need to say which views are the same and which are different in T or 12.

$\Pi_{\log} < G$
$\Pi_{\mathrm{out}} < G$ – if more than 1, which is preferred?

$D \cdot J(\Pi_{1}, \Pi_{d})$ – . . . . , Even bigger etc
pick G-stations
way to get that G exactly

=> Many subgroups of R