Production	Semantic Attachments
<S>→<Modified1>	{Modified1.sem}
<S>→<Modified2>	{Modified2.sem}
<S>→<Final>	{Final.sem}
<S>→ x	{λz. IF(IsInfo z) [] (Eqv(z, information-x))}
<Modified1$_0$>→m<Modified1$_1$>	{$\lambda y.\lambda m$.Modified1$_1$.sem(Concat(y, m)); λm.KindOf(WordOf(Modified1$_0$), Concat(m, information-x)); KindOf(WordOf(Modified1$_0$), WordOf(Modified1$_1$))}
<Modified1>→m<Modified2>	{$\lambda y.\lambda m$.Modified2.sem(Concat(y, m)); λm.KindOf(WordOf(Modified1), Concat(m, information-x)); KindOf(WordOf(Modified1), WordOf(Modified2)}
<Modified1>→m<Final>	{$\lambda y.\lambda m$.Final.sem(Concat(y, m)); λm.KindOf(WordOf(Modified1), Concat(m, information-x)); KindOf(WordOf(Modified1), WordOf(Final))}
<Modified1>→ mx	{$\lambda m.\lambda z$. KindOf(Concat(m, z), z)); λz.IF(IsInfo z) [] (Eqv(z, information-x))}
<Modified2>→a<Final>	{$\lambda y.\lambda a$. Final.sem(Concat(y, a)); λa.KindOf(WordOf(Modified2), Concat(a, information-x)); KindOf(WordOf(Modified2), WordOf(Final)); λa.Eqv(a, Concat(a, information-x))}
<Modified2>→e<Final>	{$\lambda y.\lambda e$. Final.sem(Concat(y, e)); λe.PartOf(WordOf(Modified2), e); KindOf(WordOf(Modified2), WordOf(Final)); λe.Eqv(e, Concat(e, information-x))}
<Modified2>→a<Info>	{$\lambda y.\lambda a$. Info.sem(Concat(y, a)); KindOf(WordOf(Modified2), WordOf(Info)); λa.Eqv(a, Concat(a, information-x))}
<Final>→t<Part>	{$\lambda y.\lambda t$. Part.Sem(Concat(y, t)); KindOf(WordOf(Final), WordOf(Part)); Map(λz.PartOf(Concat(z, WordOf(Part)),z))$\lambda y.\lambda$ t. SubVariant(Concat(y, t))}
<Final>→t<Info>	{$\lambda y.\lambda t$. Info.sem(Concat(y, t)); KindOf(WordOf(Final), WordOf(Info)); λt.Eqv(t,Concat(t, information-x))}
<Final>→e<Info>	{$\lambda y.\lambda e$. Info.sem(Concat(y, e)); KindOf(WordOf(Final), WordOf(Info)); λe.Eqv(e,Concat(e, information-x))}
<Final>→p	{(Map($\lambda p.\lambda z$.PartOf(p, z)))λy.SubVariant(y); $\lambda y.\lambda p$.PartOf(Concat(y,p),y)}
<Part>→<Modified1>	{λy. Modified1.sem(y)}
<Part>→<Modified2>	{λy.Modified2.sem(y)}
<Part>→<Final>	{λy.Final.sem(y)}
<Info>→x	{λz. IF(IsInfo z) [] (Eqv(z, information-x)); $\lambda z.\lambda y$. IF(IsInfo z) [] (Eqv(Concat(y, z), Concat(y, information-x))) }
<Info>→ ϵ	{}

Semantic Attachments for the Context-Free Grammar