```
SlideLeg[i_, S[SRules_List]][prims_] := ReducePrimitives[prims /. {
     Ar[i, j_] :> (Ar[i, 0] /. SRules /. 0 → j),
     Ar[j_, i] :> (Ar[0, i] /. SRules /. 0 → j),
     Y[i, j_, i, h_] :> ImClueless,
     Y[j_, i, i, h_] :> ImClueless,
     Y[i, i, j_, h_] :> Trouble,
     Y[i, j_, k_, h_] :> (Ar[i, 0] /. SRules /. {
         Ar[i, 0] :> Y[i, j, k, h],
         Y[l_, m_, 0, h1_] :> Y[l, m, k, s x[j] * h * h1]
        }),
     Y[j_, i, k_, h_] :> (Ar[i, 0] /. SRules /. {
         Ar[i, 0] :> Y[j, i, k, h],
         Y[l_, m_, 0, h1_] :> Y[l, m, k, -s x[j] * h * h1]
        }),
     Y[j_, k_, i, h_] :> (Ar[0, i] /. SRules /. {
         Ar[0, i] :> Y[j, k, i, h],
         Y[0, l_, m_, h1_] :> Y[j, k, m, s x[l] * h * h1],
         Y[l_, 0, m_, h1_] :> Y[j, k, m, -s x[l] * h * h1]
        })
    }];
Scatter[s_S][prims_] := ReducePrimitives[prims /. {
    Ar[i_, j_] :> (Ar[i, j] // SlideLeg[i, s] // SlideLeg[j, s]),
    Y[i_, j_, k_, h_] :>
     (Y[i, j, k, h] // SlideLeg[i, s] // SlideLeg[j, s] // SlideLeg[k, s])
   }]
```

On Dec 19, 2008 I decided to spilt the Ar line from the Y line in scatter routine, but the result (shown below) turned out more complicated the original, and it still needs fixing.

```
(prims_ // S[srules__Rule]) := ReducePrimitives[prims
   /. {
    Ar[i_, j_] :> (
      Distribute[Ar[Ar[i, 0], Ar[0, j]] /. {srules}]
         /. {Ar[k_, 0] :> k, Ar[0, k_] :> k}
        /. Ar[Y[k_, l_, 0, h_], m_] :> Y[k, l, m, h]
       /. {
        Ar[k_, Y[0, l_, m_, h_]] :> Y[k, l, m, h],
        Ar[k_, Y[l_, 0, m_, h_]] :> Y[l, k, m, h]
       }
     ),
    Y[i_, j_, k_, h_] :> (
      Distribute[Y[Ar[i, 0] , Ar[j, 0], Ar[0, k]] /. {srules}]
           /. {Ar[l_, 0] :> l, Ar[0, l_] :> l}
          /. {
           Y[_Y, _Y, __] :> 0,
           Y[l_, Y[i1_, j1_, 0, h1_], m_] :> Y[i1, j1, m, x[l] * h * h1],
           Y[Y[i1_, j1_, 0, h1_], l_, m_] :> Y[i1, j1, m, -x[l] * h * h1]
          }
         /. {
          Y[i1_, j1_, Y[0, m_, l_, h1_]] :> Y[i1, j1, l, -x[m] * h * h1],
          Y[i1_, j1_, Y[m_, 0, l_, h1_]] :> Y[i1, j1, l, x[m] * h * h1]
         }
        /. Y[i1_, j1_, k1_] :> Y[i1, j1, k1, h]
     )
   }
 ]

(prims_ // S[srules__Rule]) := ReducePrimitives[prims
       /. {
        Ar[i_, j_] :> Distribute[Ar[Ar[i, 0], Ar[0, j]] /. {srules}],
        Y[i_, j_, k_, h_] :> Distribute[Y[
            Ar[i, 0] , Ar[j, 0], Ar[0, k], h
           ] /. {srules}]
       }
      /. {Ar[i_, 0] :> i, Ar[0, j_] :> j}
     /. {
      Ar[Y[i_, j_, 0, h_], k_] :> Y[i, j, k, h],
      Y[_Y, _Y, __] :> 0,
      Y[i_Integer, Y[j_, k_, 0, h_], l_, h1_] :> Y[j, k, l, x[i] * h * h1],
      Y[Y[j_, k_, 0, h_], i_Integer, l_, h1_] :> Y[j, k, l, -x[i] * h * h1]
     }
    /. {
     Ar[i_, Y[0, j_, k_, h_]] :> Y[i, j, k, h],
     Ar[i_, Y[j_, 0, k_, h_]] :> Y[j, i, k, h],
     Y[i_, j_, Y[0, k_, l_, h_], h1_] :> Y[i, j, l, -x[k] * h * h1],
     Y[i_, j_, Y[k_, 0, l_, h_], h1_] :> Y[i, j, l, x[k] * h * h1]
    }
 ]
```

Brackets

```
Bracket[a_?NumberQ, _] = 0;
Bracket[_, a_?NumberQ] = 0;
Bracket[a_Plus, b_] := Bracket[#, b] & /@ a;
Bracket[a_, b_Plus] := Bracket[a, #] & /@ b;
Bracket[a_, b_ * c_] := b * Bracket[a, c] + Bracket[a, b] * c;
Bracket[a_ * b_, c_] := Bracket[a, c] * b + a * Bracket[b, c];
Bracket[Ar[i_, j_], Ar[i_, k_]] = 0;
Bracket[Ar[i_, k_], Ar[j_, k_]] := Y[i, j, k, 1];
Bracket[Ar[i_, j_], Ar[j_, k_]] := Y[i, j, k, -1];
Bracket[Ar[j_, k_], Ar[i_, j_]] := Y[i, j, k, 1];
Bracket[Ar[i_, j_], Ar[k_, l_]] = 0;
Bracket[Ar[l_, k_], Y[i_, j_, k_, h_]] := Y[i, j, k, x[l] h];
Bracket[Y[i_, j_, k_, h_], Y[i1_, j1_, k1_, h1_]] := (
   Bracket[Bracket[Ar[i, k], Y[i1, j1, k1, h * h1]], Ar[j, k]] +
    Bracket[A[i, k], Bracket[A[j, k], Y[i1, j1, k1, h * h1]]]
  );
```

Not done, and already there are too many... Many [Ar,Y] cases are missing, as well as cases where several indices coincide.

- **Just Testing**

```
(* x[i_] := x[i] = h*ToExpression["x"<>ToString[i]];
CanonicalForm[l_List] := CanonicalForm /@ l;
CanonicalForm[expr_] := Expand/@ (expr + O[h]^4)
*)
```

- **Strands Operations**

```
(*
SOp[op___Rule][S[srules___Rule]] := Module[{op1},
  op1={op} /. (i_ → j_Integer) :> (i → {j});
  expr /. {
    Ar[i_,j_] :> (Ar[i,j] /. op1),
    Y[i_,j_,k_,h_] :> Y[i/.op1, j/.op1, k/.op1, h/.
    *)
```

- **Misc, January 12 2009.**

```
hs = If[FreeQ[outs, PH], AH, PH];

AH /: MatrixExp[AH, mat_] := CanonicalForm[Map[AH, MatrixExp[mat], {2}]];
```

```
PH /: MatrixExp[PH, mat_] := Module[{target},
   target = Min[Cases[
      mat,
      Literal[SeriesData[z, 0, _, _, max_, _]] :> max,
      Infinity
     ]];
   Map[ToPH[target, PH[#]] &,
     Sum[MatrixPower[mat, k] / k!, {k, 0, target}],
     {2}] /. PH[h_] :> h
  ];
```