```
SetDirectory["c:/drorbn/projects/FastKh/"];
AppendTo[$Path, "c:/drorbn/projects/KAtlas/"];
<< \KnotTheory`
```

```
Loading KnotTheory`...
```

```
Off[General::spell]; Off[General::spell1]; $RecursionLimit = 65536;
```

```
(* BeginPackage["FastKh`",{"KnotTheory`"}] *)
```

```
Smoothing::usage =
  "Smoothing[Q^m a1 a2 ...] denotes a smoothing made of components a1,
    a2, etc., where each ai is either a Loop[k] or a P[i,j][min],
    and where m is an overall degree shift. Sometimes the head
    `Smoothing' is suppressed and the head `Times' is used instead.";

Cobordism::usage =
  "\n  Cobordism[top, bot, d1 d2 ...] denotes a cobordism with top
    top, bottom\n  bot, and extra decorations d1, d2, ....
    Each di is either a tdot[k], or\n  a bdot[k]. If no
    decorations are present, this is a `cobordism prototype'.\n";

HCLaw::usage =
  "\n  HCLaw[cob1, cob2] takes two cobordism prototypes and returns a pair\n
    {cob, law} consisting of the output cobordism resulting from the
    horizontal\n  composition of cob1 and cob2 and a rule containing
    the decoration transfer\n  law for the input decorations.\n";

{bdot, tdot};
```

```
(*Begin["`FastKh`"]*)
```

```
bdot[_]^_ ^= 0; tdot[_]^_ ^= 0;
```

```
EquivalenceClasses[l_List] := Fold[
   (
     pos = First /@ Position[#1, #2];
     Append[Delete[#1, List /@ pos], Union @@ (#1[[pos]])]
    ) &,
   l, Union @@ l
  ];

DotRule[top_, bot_] := Flatten[Cases[
    DeleteCases[
     EquivalenceClasses[Join[
       Cases[{top}, P[i_, j_][m_] :> {z@i, z@j, tdot@m}, Infinity],
       Cases[{bot}, P[i_, j_][m_] :> {z@i, z@j, bdot@m}, Infinity]
      ]],
     _z, {2}
    ],
    l_List :> ((# -> First[l]) & /@ Rest[l])
   ]];

Deg[Cobordism[top_Smoothing, bot_Smoothing, decors_]] :=
  Plus[
   Length[Union[Last /@ DotRule[top, bot]]],
   Count[{top}, _Loop, Infinity],
   Count[{bot}, _Loop, Infinity],
   -Count[{top}, P[__][_], Infinity],
   -2 * Exponent[First@Flatten[{decors}] /. {_bdot -> Q, _tdot -> Q}, Q],
   Exponent[First@bot, Q] - Exponent[First@top, Q]
  ];

HCLaw[
    Cobordism[top1_Smoothing, bot1_Smoothing],
    Cobordism[top2_Smoothing, bot2_Smoothing]
   ] /; MemberQ[{top1, bot1, top2, bot2}, Q, Infinity] := MapAt[
   (Q^Exponent[Times @@ bot1, Q] * Q^Exponent[Times @@ bot2, Q]) &,
   MapAt[
    (Q^Exponent[Times @@ top1, Q] * Q^Exponent[Times @@ top2, Q]) &,
    HCLaw[Cobordism[top1, bot1] /. Q -> 1, Cobordism[top2, bot2] /. Q -> 1],
    {1, 1, 1}
   ],
   {1, 2, 1}
  ];
HCLaw[
    Cobordism[top1_Smoothing, bot1_Smoothing],
    Cobordism[top2_Smoothing, bot2_Smoothing]
   ] /; FreeQ[{top1, bot1, top2, bot2}, Q] :=
  HCLaw[Cobordism[top1, bot1], Cobordism[top2, bot2]] = Module[
    {top, bot, dots, dr, decors, law, to, cob1, cob2, cob, deg1, deg2, deg},
    top = HC[top1, top2];
    bot = HC[bot1, bot2];
```

```
    dots = Union[
      Last /@ DotRule[top, bot],
      Cases[{top}, Loop[m_] :> tdot[m], Infinity],
      Cases[{bot}, Loop[m_] :> bdot[m], Infinity]
     ];
    dr = DotRule[top1 top2, bot1 bot2];
    decors = Expand[Times @@ MapThread[
        If[#1 === #2, 1, #1 + #2] &,
        {dots, dots /. dr}
       ]];
    law = Union[
      Last /@ DotRule[top1, bot1], Last /@ DotRule[top2, bot2]
     ];
    law = DeleteCases[
       Thread[to[law, law /. dr]],
       to[m_, m_]
      ] /. to -> Rule;
    (*
    deg1 = Deg[cob1 = Cobordism[top1,bot1, 1]];
    deg2 = Deg[cob2 = Cobordism[top2,bot2, 1]];
    deg = Deg[cob = Cobordism[top,bot,decors]];
    If[deg1 + deg2 =!= deg && Count[{deg1, deg2, deg}, Infinity] == 0,
     Print["Degree mismatch at HCLaw:"];
     Print["cob1 = ", cob1];
     Print["cob2 = ", cob2];
     Print["cob = ", cob];
     Print["Degrees: ", {deg1, deg2, deg}]
    ];
    *)
{Cobordism[top, bot, decors], law}
   ];
HCLaw[a_, b_] := Print["HCLaw: ", a, " and ", b]

HC[0, _] = HC[_, 0] = 0;
HC[Smoothing[s1_], Smoothing[s2_]] :=
  Smoothing[s1 s2 //. P[i_, j_][m_] P[j_, k_][n_] :> P[i, k][Min[m, n]] /.
    {P[i_, j_][m_]^2 :> Loop[m], P[i_, i_][m_] :> Loop[m]}];

HC[n1_. * e[t1__] * s1_Smoothing, n2_. * e[t2__] * s2_Smoothing] :=
  n1 n2 e[t1, t2] HC[s1, s2];

HC[
   Cobordism[top1_Smoothing, bot1_Smoothing, ds1_],
   Cobordism[top2_Smoothing, bot2_Smoothing, ds2_]
  ] := Module[
   {cob, law, cob1, cob2, deg1, deg2},
   {cob, law} = HCLaw[
     Cobordism[top1, bot1], Cobordism[top2, bot2]
    ];
```

```
    cob = MapAt[Expand[(ds1 ds2 /. law) * #] &, cob, 3];
    (*
    deg1 = Deg[cob1 = Cobordism[top1,bot1, ds1]];
    deg2 = Deg[cob2 = Cobordism[top2,bot2, ds2]];
    deg = Deg[cob];
    If[deg1 + deg2 =!= deg && Count[{deg1, deg2, deg}, Infinity] == 0,
     Print["Degree mismatch at HC:"];
     Print["cob1 = ", cob1];
     Print["cob2 = ", cob2];
     Print["cob = ", cob];
     Print["Degrees: ", {deg1, deg2, deg}]
    ];
    *)
cob
  ];

HC[a_Plus, b_] := HC[#, b] & /@ a;
HC[a_, b_Plus] := HC[a, #] & /@ b;

HC[Morphism[top_, bot_, a_ + b_], s_] := Plus[
   HC[Morphism[top, bot, a], s],
   HC[Morphism[top, bot, b], s]
  ];
HC[Morphism[top_, bot_, MM[e[i___], e[j___], mat_]], e[k___] * s_Smoothing] :=
  Module[
   {cob, law},
   {cob, law} = HCLaw[
     Cobordism[Coefficient[top, e[i]], Coefficient[bot, e[j]]],
     Cobordism[s, s]
    ];
   MM[e[i, k], e[j, k], Expand[Last[cob] * (mat /. law)]]
  ];

HC[s_, Morphism[top_, bot_, a_ + b_]] := Plus[
   HC[s, Morphism[top, bot, a]],
   HC[s, Morphism[top, bot, b]]
  ];
HC[e[k___] * s_Smoothing, Morphism[top_, bot_, MM[e[i___], e[j___], mat_]]] :=
  Module[
   {cob, law},
   {cob, law} = HCLaw[
     Cobordism[s, s],
     Cobordism[Coefficient[top, e[i]], Coefficient[bot, e[j]]]
    ];
   MM[e[k, i], e[k, j], Expand[Last[cob] * (mat /. law)]]
  ];

HC[
```

```
 Kom[f1_, obs1_, mos1_],
 Kom[f2_, obs2_, mos2_]
] := Module[
 {l1, l2, k, j1, j2, obs, morph, mos, rule},
 l1 = Length[obs1] - 1; l2 = Length[obs2] - 1;
 obs = Objects @@ Table[
    Plus @@ Table[
      j2 = k - j1;
      HC[obs1[[1 + j1]], obs2[[1 + j2]]] /. e[t__] :> e[t, j1],
      {j1, Max[0, k - l2], Min[l1, k]}
     ],
    {k, 0, l1 + l2}
   ];
 mos = Morphisms @@ Table[
    Plus @@ Table[
      j2 = k - j1;
      Plus[
       If[1 + j1 > l1 || mos1[[1 + j1]] === 0 || obs2[[1 + j2]] === 0,
        0,
        HC[
          Morphism[obs1[[1 + j1]], obs1[[2 + j1]], mos1[[1 + j1]]],
          obs2[[1 + j2]]
         ] /. MM[e[t1__], e[t2__], mm_] :> MM[e[t1, j1], e[t2, j1 + 1], mm]
       ],
       If[1 + j2 > l2 || obs1[[1 + j1]] === 0 || mos2[[1 + j2]] === 0,
        0,
        HC[
          obs1[[1 + j1]],
          Morphism[obs2[[1 + j2]], obs2[[2 + j2]], mos2[[1 + j2]]]
         ] /. MM[e[t1__], e[t2__], mm_] :>
          MM[e[t1, j1], e[t2, j1], Expand[(-1)^j1 * mm]]
       ]
      ],
      {j1, Max[0, k - l2], Min[l1, k]}
     ],
    {k, 0, l1 + l2 - 1}
   ];
 ReTag[Kom[f1 + f2, obs, mos]]
];
```

```
ReTag[kom_Kom] := Module[
  {f, obs, mos, l},
  {f, obs, mos} = List @@ kom;
  l = Length[obs] - 1;
  Do[
   rule = Union[Cases[{obs[[1 + k]]}, _e, Infinity]];
   rule = Thread[Rule[rule, e /@ Range[Length[rule]]]];
   obs[[1 + k]] = obs[[1 + k]] /. rule;
   If[k < l,
    mos[[1 + k]] = mos[[1 + k]] /. MM[e1_, e2_, mm_] :> MM[e1 /. rule, e2, mm]
   ];
   If[k > 0,
    mos[[k]] = mos[[k]] /. MM[e1_, e2_, mm_] :> MM[e1, e2 /. rule, mm]
   ],
   {k, 0, l}
  ];
  Kom[f, obs, mos]
 ]
```

```
Clear[VCLaw];
VCLaw[
   Cobordism[top_Smoothing, mid_Smoothing],
   Cobordism[mid_Smoothing, bot_Smoothing]
  ] := VCLaw[Cobordism[top, mid], Cobordism[mid, bot]] = Module[
    {decors, law1, law2, dots, dots1, dots2, dr1, dr2, dr, to},
    {law1, law2} = {{}, {}};
    decors = Times @@ Cases[
       {mid},
       Loop[m_] :> (
         AppendTo[law1, bdot[m] → mdot[m]];
         AppendTo[law2, tdot[m] → mdot[m]];
         mdot[m]
        ),
       Infinity
      ];
    dots = Union[Last /@ DotRule[top, bot]];
    dots1 = Union[Last /@ (dr1 = DotRule[top, mid] /. bdot → mdot)];
    dots2 = Union[Last /@ (dr2 = DotRule[mid, bot] /. tdot → mdot)];
    dr = Flatten[Cases[
       EquivalenceClasses[Join[List @@@ dr1, List @@@ dr2]],
       l_List :> ((# → First[l]) & /@ Rest[l])
      ]];
    decors *= Expand[Times @@ MapThread[
        If[#1 === #2, 1, #1 + #2] &,
        {dots, dots /. dr}
       ]];
    law1 = Join[law1,
      DeleteCases[
        Thread[to[dots1, dots1 /. dr]] /. mdot → bdot,
        to[m_, m_]
       ] /. to → Rule
     ];
    law2 = Join[law2,
      DeleteCases[
        Thread[to[dots2, dots2 /. dr]],
        to[m_, m_]
       ] /. to → Rule
     ];
    {law1, law2, decors}
   ];
```

```
VC[a_, b_, c__] := VC[a, VC[b, c]];
VC[
   Cobordism[top_Smoothing, mid_Smoothing, ds1_],
   Cobordism[mid_Smoothing, bot_Smoothing, ds2_]
  ] := Module[
   {law1, law2, decor, cob, cob1, cob2, deg, deg1, deg2},
   {law1, law2, decor} = VCLaw[Cobordism[top, mid], Cobordism[mid, bot]];
   cob = Cobordism[top, bot,
     Expand[decor * (ds1 /. law1) * (ds2 /. law2)] /. (_mdot)^2 → 1 /. (_mdot → 0)
    ];
   (*
   deg1 = Deg[cob1 = Cobordism[top,mid, ds1]];
   deg2 = Deg[cob2 = Cobordism[mid,bot, ds2]];
   deg = Deg[cob];
   If[deg1 + deg2 =!= deg && Count[{deg1, deg2, deg}, Infinity] == 0,
    Print["Degree mismatch at VC:"];
    Print["cob1 = ", cob1];
    Print["cob2 = ", cob2];
    Print["cob = ", cob];
    Print["Degrees: ", {deg1, deg2, deg}]
   ];
   *)
cob
  ];
```

```
DeLoop[kom_Kom] := Module[
   {f, obs, mos, l, dot},
   {f, obs, mos} = List @@ kom;
   l = Length[obs] - 1;
   Do[
    obs[[1 + k]] = obs[[1 + k]] //. e[i___] Smoothing[Loop[j_] * rest_.] :> (
        If[k > 0, mos[[k]] = mos[[k]] /. MM[e[l___], e[i], mat_] :> Plus[
             MM[e[l], e[i, -1],
              Expand[dot[j] * mat] /. bdot[j] dot[j] → 1 /. dot[j] → 0
             ],
             MM[e[l], e[i, 1],
              mat /. bdot[j] → 0
             ]
            ]];
        If[k < l, mos[[1 + k]] = mos[[1 + k]] /. MM[e[i], e[l___], mat_] :> Plus[
             MM[e[i, -1], e[l],
              mat /. tdot[j] → 0
             ],
             MM[e[i, 1], e[l],
              Expand[dot[j] * mat] /. tdot[j] dot[j] → 1 /. dot[j] → 0
             ]
            ]];
        e[i, -1] Smoothing[rest / Q] + e[i, 1] Smoothing[rest * Q]
       ),
    {k, 0, l}
   ];
   ReTag[Kom[f, obs, mos] /. MM[_, _, {{0}}] → 0]
  ];

Contract[kom_Kom] := Module[
   {
    f, obs, mos, l, k, e2s0,
    e2s1, s2b, b, e2b0, e2b1, killed0, killed1, done, mok
   },
   {f, obs, mos} = List @@ kom;
   l = Length[obs] - 1;
   Do[
    e2s0 = Cases[{obs[[1 + k]]}, i_e * s_Smoothing :> (i → s), Infinity];
    e2s1 = Cases[{obs[[1 + k + 1]]}, i_e * s_Smoothing :> (i → s), Infinity];
    s2b = Union[Union[Last /@ e2s0, Last /@ e2s1] /. P[j__][m_] :> P[j]];
    s2b = Thread[Rule[s2b, b /@ Range[Length[s2b]]]];
    e2b0 = e2s0 /. P[j__][m_] :> P[j] /. s2b;
    e2b1 = e2s1 /. P[j__][m_] :> P[j] /. s2b;
    killed0 = killed1 = {}; done = False;
    While[! done,
     done = True;
     mok = mos[[1 + k]];
```

```
      Cases[
       {mok},
       MM[i_e, j_e, {{r_?NumberQ}}] /;
         ((i /. e2b0) === (j /. e2b1))
:> (
         mok = Plus[
           mok /. {MM[i, _, _] → 0, MM[_, j, _] → 0},
           Expand[-Plus @@ Flatten[Outer[
                 Function[{M1, M2},
                  MM[M1[[1]], M2[[2]], Last[VC[
                     Cobordism[M1[[1]] /. e2s0, j /. e2s1, M1[[3, 1, 1]]],
                     Cobordism[j /. e2s1, i /. e2s0, {{1 / r}}],
                     Cobordism[i /. e2s0, M2[[2]] /. e2s1, M2[[3, 1, 1]]]
                    ]]
                  ]
                 ],
                 Cases[{mok}, MM[i1_e, j, mm1_] /; i1 =!= i, Infinity],
                 Cases[{mok}, MM[i, j1_e, mm2_] /; j1 =!= j, Infinity]
                ]]]
          ];
         mos[[1 + k]] =
          (((mok //. a_ * MM[i1_, j1_, mm_] :> MM[i1, j1, Expand[a * mm]])
 //. MM[i1_, j1_, mm1_] + MM[i1_, j1_, mm2_] :> MM[i1, j1, mm1 + mm2])
 /. MM[_, _, {{0}}] → 0);
         done = False;
         AppendTo[killed0, i]; AppendTo[killed1, j]
        ),
       Infinity, 1]
     ];
     obs[[1 + k]] = obs[[1 + k]] /. ((# → 0) & /@ killed0);
     obs[[1 + k + 1]] = obs[[1 + k + 1]] /.  ;;
If[k > 0,
        mos[[1 + k - 1]] = mos[[1 + k - 1]] /.
          MM[i_e, j_e, mm_] /; MemberQ[killed0, j] :> 0
       ];
      If[k < l - 1,
       mos[[1 + k + 1]] = mos[[1 + k + 1]] /.
         MM[i_e, j_e, mm_] /; MemberQ[killed1, i] :> 0
      ],
      {k, 0, l - 1}
     ];
     ReTag[Kom[f, obs, mos]]
    ];
```

```
FindThinPD[L_] := FindThinPD[PD[L]];
FindThinPD[pd_PD] := Module[
  {thin, todo, front, F, fronts , pos},
  thin = ThinPD[];
  todo = List @@ pd;
  front = F[];
  While[Length[todo] > 0,
   fronts = (
       # * front /. {
          X[l1___, m_, r1___] F[l2___, m_, r2___] :>
           Flatten[F[l2, Reverse[F[r1, l1]], r2]],
          F[f___] X[x___] :> Flatten[F[f, Reverse[F[x]]]]
         }
 //. F[l___, m_, m_, r___] :> F[l, r]
      ) & /@ todo;
   {pos} = Ordering[fronts, 1];
   AppendTo[thin, todo[[pos]]];
   front = fronts[[pos]]; Print[todo[[pos]], " ", front];
   todo = Delete[todo, pos];
  ];
  thin
 ]
```

```
KhComplex[L_] := KhComplex[FindThinPD[L]]
KhComplex[X[i_, j_, k_, l_]] /; (j - l == 1 || l - j > 1) := Kom[0, (* + xing *)
Objects[
      e[1] Smoothing[Q P[i, j] P[k, l]],
      e[1] Smoothing[Q^2 P[i, l] P[j, k]]
     ] /. P[m_, n_] :> P[m, n][Min[m, n]],
    Morphisms[MM[e[1], e[1], {{1}}]]
  ];
KhComplex[X[i_, j_, k_, l_]] /; (l - j == 1 || j - l > 1) := Kom[-1, (* - xing *)
Objects[
      e[1] Smoothing[Q^(-2) P[i, j] P[k, l]],
      e[1] Smoothing[Q^(-1) P[i, l] P[j, k]]
     ] /. P[m_, n_] :> P[m, n][Min[m, n]],
    Morphisms[MM[e[1], e[1], {{1}}]]
  ];
KhComplex[tpd_ThinPD] /; (Length[tpd] > 1) := Module[
  {kom},
  kom = KhComplex[First@tpd];
  Print["Starting with ", First@tpd];
  Do[
   kom = HC[kom, KhComplex[tpd[[i]]]];
   kom = DeLoop[kom];
   kom = Contract[kom];
   Print[i, ": Fused ", tpd[[i]], ", sizes are ", Length /@ kom[[2]]],
   {i, 2, Length[tpd]}
  ];
  kom
 ]

FastKh[kom_Kom] := Module[
   {f, obs, mos},
   {f, obs, mos} = List @@ kom;
   If[Union[List @@ mos] =!= {0}, Error,
    Plus @@ Expand[t^(f - 1) * t^Range[Length[obs]] * (
         List @@ obs /. e[i_] Smoothing[s_] :> s /. Q -> q
        )]
   ]
  ];
FastKh[L_] := FastKh[KhComplex[L]]

(* End[]; EndPackage[] *)
```