ConciseFastKh on June 23, 2013

June-23-13
7:40 AM

$Q[\rho] \to q^\rho$

Pensieve header: A concise implementation of the FastKh algorithm.

```
<< KnotTheory`

Loading KnotTheory` version of February 5, 2013, 3:48:46.4762.
Read more at http://katlas.org/wiki/KnotTheory.

SetAttributes[{P, S}, Orderless];
dot /: dot[_]^k_ := 0;
(σ_S)[i_] := First@Cases[σ, P[i, j_] :> j]; {Memorize?}

(* ECP for "Equivalence Class Projection" *)
ECP[λ_List] := Module[{ρ, ec},
    ec = Fold[
        (ρ = First /@ Position[#1, #2];
          Append[Delete[#1, List /@ ρ], Union @@ (#1[[ρ]])]) &,
        λ, Union @@ λ
    ] ~~// SortBy[#, First] &~~;                          verify?
   Union @@ Replace[ec, c_ :> ((# → First[c]) & /@ c), {1}]
  ];
ECP[λ__S] := ECP[Join[λ] /. S | P → List];
(* ECP for "Equivalence Class Representatives" *)
ECR[λ__] := Union[Last /@ ECP[λ]];

VCLaw[β_S, μ_S, τ_S] := VCLaw[β, μ, τ] = Module[
    {p, ins1, ins2, outs, χs, ~~χ1~~, h, law1, law2, dec},
    p = ECP[β, μ, τ];
    ins1 = ECR[β, μ]; ins2 = ECR[μ, τ]; outs = ECR[β, τ];
    χs = Times @@ (h /@ Join[ins1, ins2, outs] /. p) / PowerExpand[(Times @@ (h /@ (Last /@ p)))^(1/2)];
    dec = χs /. h[i_]^χ_. :> (2 dot[i])^((2-χ)/2);
    dec *= Times @@ MapThread[If[#1 == #2, 1, dot[#1] + dot[#2]] &,
        {outs, outs /. p}];
    law1 = dot /@ ins1; law1 = Thread[law1 → (law1 /. p)];
    law2 = dot /@ ins2; law2 = Thread[law2 → (law2 /. p)];
    {law1, law2, Expand[dec]}
  ];

VC[Cob[β_S, μ_S, dots1_], Cob[μ_S, τ_S, dots2_]] := Module[
    {law1, law2, dec},
    {law1, law2, dec} = VCLaw[β, μ, τ];
    ~~Cob[β, τ,~~ Expand[dec * (dots1 /. law1) (dots2 /. law2)]~~]~~
  ];

m[i_, j_][σ_S] := Which[
    σ[i] ≠ j, Append[DeleteCases[σ, P[i, _] | P[_, j]], P[σ[i], σ[j]]],
    σ[i] == j, DeleteCases[σ, P[i, j]]
  ];
m[i_, j_][Q[k_] * σ_S] := m[i, j][σ] * If[σ[i] ≠ j, {Q[k]}, {Q[k + 1], Q[k - 1]}];
```

```
m[i_, j_][Cob[β_S, τ_S, dots_]] := Module[
  {nβ, nτ, p, ijdot, ndots, x},
  {nβ, nτ} = m[i, j] /@ {β, τ};
  p = ECP[β, τ];
  ijdot = dot[Min[i, j]];
  ndots = Which[
    β[i] ≠ j && τ[i] ≠ j, {{If[(i /. p) ≠ (j /. p), 1, dot[β[i]] + dot[τ[i]]]}},
    β[i] == j && τ[i] ≠ j, {{1, ijdot}},
    β[i] ≠ j && τ[i] == j, {{ijdot}, {1}},
    β[i] == j && τ[i] == j, ( ijdot   0
                               1   ijdot )
  ];
  ndots = Expand[dots * ndots] /. dot[k_] :>
    dot[k /. {i → β[i], j → β[j]} /. {i → τ[i], j → τ[j]} /. ECP[nβ, nτ]];
  If[β[i] == j && τ[i] == j, Coefficient[ndots /. ijdot → x, x], ndots]
];

Q /: Q[i_] Q[j_] := Q[i + j];
S /: s1_S * s2_S := Join[s1, s2];

Kom /: Kom[chains_, ds_] * Cob[β_, τ_, 1] := Module[{L, ρ, nchains, d, k},
  L = Length[chains]; ρ_k_ := ρ_k = Length[chains[[k]]]; ρ_0 = ρ_{L+1} = 0;
  Kom[
   nchains = MapThread[Join, List @@@ {
      Append[chains /. σ_S :> β σ, {}],
      Prepend[chains /. σ_S :> τ σ, {}]
     }],
   Table[
    If[(ρ_k + ρ_{k-1}) (ρ_{k+1} + ρ_k) == 0, 0,
     (*else*) d = Table[0, {ρ_{k+1} + ρ_k}, {ρ_k + ρ_{k-1}}];
     If[k ≤ L && ρ_k ρ_{k+1} ≠ 0, d[[1 ;; ρ_{k+1}, 1 ;; ρ_k]] = ds[[k]]];
     If[k ≤ L && ρ_k ≠ 0, d[[ρ_{k+1} + 1 ;; ρ_{k+1} + ρ_k, 1 ;; ρ_k]] = (-1)^k IdentityMatrix[ρ_k]];
     If[k > 1 && ρ_{k-1} ρ_k ≠ 0, d[[ρ_{k+1} + 1 ;; ρ_{k+1} + ρ_k, ρ_k + 1 ;; ρ_k + ρ_{k-1}]] = ds[[k - 1]]];
     d
    ], {k, L}
   ]
  ]
 ]

Kom /: Show[Kom[chains_, ds_]] :=
  MatrixForm[{ColumnForm /@ chains, MatrixForm /@ Append[ds, 0]}];
```

use Join

move to "utilities"

```
m[i_, j_][Kom[chains_, ds_]] := Kom[
  Flatten /@ Map[m[i, j], chains, {2}],
  Table[
   If[Length[chains[[k]]] == 0 || Length[chains[[k + 1]]] == 0, 0,
    (* else *) Table[
     m[i, j][Cob[chains[[k, b]] /. _Q → 1,
       chains[[k + 1, a]] /. _Q → 1, ds[[k, a, b]]]],
     {a, Length[chains[[k + 1]]]}, {b, Length[chains[[k]]]}
     ] // ArrayFlatten
    ],
   {k, Length[ds]}]
  ];
```

"cs"

change to For loop

change to a double For loop, with —a x —b in middle & no break.

cut 2 lines

```
Contract[kom_Kom] := Module[{chains, ds, L, done, ϕ, γδ},
  {chains, ds} = List @@ kom;
  L = Length[ds];
  Do[ (* {k,L} *)
   done = False; While[!done,
    done = True;
    Do[ (* {a, Length[chains[[k+1]]]}, {b, Length[chains[[k]]]} *)
     If[NumberQ[ϕ = ds[[k, a, b]]] && ϕ ≠ 0 && chains[[k + 1, a]] == chains[[k, b]],
      done = False;
      If[Length[chains[[k]]] > 1 && Length[chains[[k + 1]]] > 1,
       γδ = Table[
         VC[
           Cob[chains[[k, d]], chains[[k + 1, a]], ds[[k, a, d]]] /. _Q → 1,
           Cob[chains[[k, b]], chains[[k + 1, c]], ds[[k, c, b]]] /. _Q → 1
           ][[3]],
         {c, Length[chains[[k + 1]]]}, {d, Length[chains[[k]]]}
         ];
       ds[[k]] = Expand[Drop[ds[[k]] - ϕ^-1 γδ, {a}, {b}]],
       (* else *) ds[[k]] = 0
       ];
      chains[[k]] = Drop[chains[[k]], {b}];
      chains[[k + 1]] = Drop[chains[[k + 1]], {a}];
      If[k > 1, ds[[k - 1]] = If[ds[[k - 1]] === 0, 0, Drop[ds[[k - 1]], {b}]]];
      If[k < L, ds[[k + 1]] = If[ds[[k + 1]] === 0, 0, Drop[ds[[k + 1]], {}, {a}]]];
      Break[]
      ],
     {a, Length[chains[[k + 1]]]}, {b, Length[chains[[k]]]}
     ]
    ],
   {k, L}
   ];
  Kom[chains, ds]
  ];
```

use ∩?

```
CFKh[L_] := Module[
  {pd = PD[L], kom = Kom[{{S[]}}, {}], inside = {}, tp = 0, pos},
  While[Length[pd] > 0,
   pos = Last[Ordering[(Length[Intersection[List @@ #, inside]]) & /@ pd]];
   kom = kom ⋆ (pd[[pos]] /. {
        X[i_, j_, k_, l_] /; (j - l == 1 || l - j > 1) ⧴
         Cob[Q[1] S[P[-i, j], P[k, -l]], Q[2] S[P[-i, -l], P[j, k]], 1],
        X[i_, j_, k_, l_] /; (l - j == 1 || j - l > 1) ⧴
         (--tp; Cob[Q[-2] S[P[-i, -j], P[k, l]], Q[-1] S[P[-i, l], P[-j, k]], 1])
       });
   (kom = Contract[kom // m[#, -#]]) & /@ Intersection[List @@ pd[[pos]], inside];
   inside = Union[inside, List @@ pd[[pos]]];
   pd = Drop[pd, {pos}];
  ];
  Expand[t^(tp-1+Range[Length[First[kom]]]).(List @@ Plus @@@ First @ kom) /.
    {Q[qp_] ⧴ q^qp, S[] → 1}]
]
```

use ∩?

use ∪?

Add a computation.

Consider standardizing smoothing labels.

Consider dot[i] ⟶ •ᵢ