NewtonGF[BoltzmannExpectedSize] - compute the expected size of the structures produced by a Boltzmann sampler

Calling Sequence

BoltzmannExpectedSize(**Sys**, **labelling**)

Parameters

`Sys` – set of equations; a grammar in the combstruct syntax

`labelling` – one of labelled, labeled, unlabelled, unlabeled, as in combstruct

Description

- The **BoltzmannExpectedSize** command returns a procedure that takes as input the Boltzmann parameter and returns the expected size of the structures produced by the Boltzmann sampler for the system.
- It is computed using **NewtonGF**[**NumericalNewtonIteration**] for the system and its derivative.
- This command is part of the **NewtonGF** package, so it can be used in the form **BoltzmannExpectedSize**(..) only after executing the command **with(NewtonGF)**. However, it can always be accessed through the long form of the command by using **NewtonGF**[**BoltzmannExpectedSize**](..).

Examples

```
> with(NewtonGF);
```

$$[\mathit{BoltzmannExpectedSize}, \mathit{BoltzmannParameter}, \mathit{GFSeries}, \mathit{NumericalNewtonIteration}, \mathit{Radius}, \mathit{SeriesNewtonIteration}] \quad \textbf{(2.1)}$$

A grammar for series-parallel circuits.

```
> circuit:={C=Union(P,S,R), P=Set(Union(S,R),card>=2), S=Set
  (Union(P,R),card>=2), R=Atom};
```

$$\mathit{circuit} := \{C = \mathit{Union}(P, S, R), P = \mathit{Set}(\mathit{Union}(S, R), 2 \le \mathit{card}), R = \mathit{Atom}, S = \mathit{Set}(\mathit{Union}(P, R), 2 \le \mathit{card})\} \quad \textbf{(2.2)}$$

Here are the corresponding equations over generating functions:
combstruct[gfeqns](circuit, labeled, z);

$$[C(z) = P(z) + S(z) + z, P(z) = \mathrm{e}^{S(z) + z} - 1 - S(z) - z, R(z) = z, S(z) = \mathrm{e}^{P(z) + z} - 1 - P(z) - z]$$

```
> expected_size:=BoltzmannExpectedSize(circuit, labeled);
```

$$\mathit{expected\_size} := \textbf{proc}(x) \ \ldots \ \textbf{end proc} \quad \textbf{(2.3)}$$

```
> expected_size(0.1);
```

$$[E(C) = 1.121189318, E(P) = 2.166019224, E(R) = 1.000000000, E(S) = 2.166019224] \quad \textbf{(2.4)}$$

The radius of convergence of the generating function.

```
> rho:=Radius(circuit,labeled);
```

$$\rho := 0.3862943611 \quad \textbf{(2.5)}$$

Computing the expected size near the radius.

```
> expected_size(0.3862943);
```

(2.6)

$$[E(C) = 1563.015705, E(P) = 2547.012203, E(R) = 1.000000000, E(S) = 2547.012203] \tag{2.6}$$

```
> expected_size(rho);
```

$$[E(C) = 82775.01189, E(P) = 134877.5131, E(R) = 1.000000000, E(S) = 134877.5131] \tag{2.7}$$

And with unlabelled circuits.

```
> expected_size_unlabelled:=BoltzmannExpectedSize(circuit,
  unlabeled);
```

$$\mathit{expected\_size\_unlabelled} := \textbf{proc}(x)\ \ldots\ \textbf{end proc} \tag{2.8}$$

```
> expected_size_unlabelled(0.1);
```

$$[E(C) = 1.258938315, E(P) = 2.281140450, E(R) = 1.000000000, E(S) = 2.281140450] \tag{2.9}$$

```
> rho:=Radius(circuit,unlabeled);
```

$$\rho := 0.2808326670 \tag{2.10}$$

```
> expected_size_unlabelled(0.28083);
```

$$[E(C) = 238.1170934, E(P) = 331.2931422, E(R) = 1.000000000, E(S) = 331.2931422] \tag{2.11}$$

This procedure does no answer for a value outside the radius of convergence of the generating function.

```
> expected_size_unlabelled(0.29);
Error, (in oracle) Newton iteration does not seem to converge
```

See Also

combstruct[gfseries], NewtonGF[NumericalNewtonIteration], NewtonGF[Radius], NewtonGF[BoltzmannParameter], NewtonGF